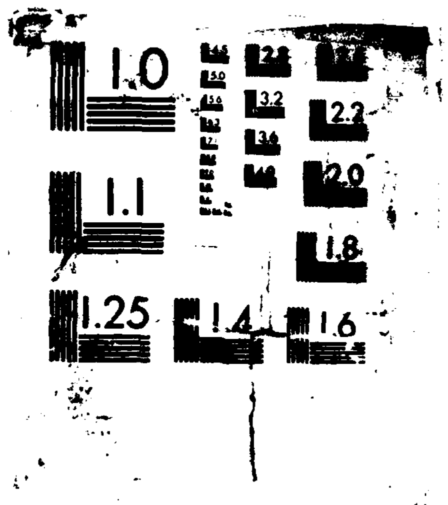
AD-A172 423   MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS   1/2
FOR FLIGHT CONTROL..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI..   J J CERRA
UNCLASSIFIED   JUN 86 AFIT/GAE/AA-86J-2   F/G 1/3   NL



MICROCOPY RESOLUTION TEST CHART

MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS FOR FLIGHT CONTROL DEVELOPMENT

THESIS

John J. Cerra II
Captain, USAF

AFIT/GAE/AA/86J-2

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

(1)

AFIT/GAE/AA/86J-2

MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS FOR FLIGHT CONTROL DEVELOPMENT

THESIS

John J. Cerra II
Captain, USAF

AFIT/GAE/AA/86J-2

1

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD-A172423

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY 2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GAE/AA/86J-2	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL *(If applicable)* AFIT/ENY	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS *(City, State and ZIP Code)* Air Force Institute of Technology Wright-Patterson AFB, OH 45433	7b. ADDRESS *(City, State and ZIP Code)*
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Flight Dynamics Laboratory 8b. OFFICE SYMBOL *(If applicable)* AFWAL/FIBR	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State and ZIP Code)* Wright-Patterson AFB, OH 45433	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE *(Include Security Classification)* See Box 19	

12. PERSONAL AUTHOR(S)
John J. Cerra II, B.S., Captain, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Yr., Mo., Day)*	15. PAGE COUNT
MS Thesis	FROM ____ TO ____	1986 June	117

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.
20	11	
01	03	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Aeroelasticity, Airplane models, Aerodynamic Stability, Flight Control Systems, Theses.

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS FOR FLIGHT CONTROL DEVELOPMENT

Thesis Chairman: Dr. Robert A. Calico
Professor of Aeronautical Engineering

Approved for public release: IAW AFR 190-1.
[illegible] 19 Sept 86
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Robert A. Calico	22b. TELEPHONE NUMBER *(Include Area Code)* (513) 255-4476 22c. OFFICE SYMBOL AFIT/ENY

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

Abstract

The purpose of this effort was to provide a method of developing a linear model of an elastic aircraft. The model provides the capability to analyze the coupling between the rigid and elastic motion of the aircraft. The method developed in this effort obtains stability derivatives directly from unsteady aerodynamic forces. This results in a state-space model whose states are just the normal aircraft states and rates, the structural coordinates and rates, and the control surface positions and rates. Using a representation of the YF-17 wind tunnel flutter model, it was demonstrated that the methodology developed predicted the required dynamics to make this a viable method of modelling rigid-body and flutter behavior of the model. Flutter control laws were designed for motion about an equilibrium condition represented by a velocity 20% above the flutter velocity. Both classical and modern techniques yielded acceptable control laws. The control laws were also analyzed at off design conditions to check robustness. Keywords [illegible] 18

AFIT/GAE/AA/86J-2

MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS FOR FLIGHT CONTROL DEVELOPMENT

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Aeronautical Engineering

John J. Cerra II, B.S.

Captain, USAF

June 1986

Preface

The interaction between the flight control system, structural dynamics, and aerodynamics of aircraft has become a major concern to aircraft designers. There has been considerable effort to develop an accurate method of modelling such interactions. This effort develops a simple method to create an accurate model of this interaction. This model can then be used for stability and control analysis including the effects of structural dynamics.

I wish to thank my advisor, Dr. Robert A. Calico for his invaluable guidance and assistance during this research effort. I would also like to thank my thesis committee members, Dr. Peter J. Torvik and Maj. Lanson Hudson for their helpful comments and thorough editing of this document. In addition, I would like to thank my sponsor, Dr. Thomas E. Noll for his guidance on applying aeroelastic concepts and for the use of his mathematical model of the YF-17. My thanks also go to Mr. Maxwell Blair who helped me use ADAM, and 1Lt. William Blake who helped run Digital DATCOM. Finally, I wish to thank my wife Colleen for her understanding and support during this work.

Table of Contents

List of Figures

List of Tables

Nomenclature

Scalars

a_i - components of a skew symmetric matrix

b - reference semi-chord length

$\bar{c}$ - reference chord length

D_i - denominator coefficient of the Pade polynomial

F_i - ith generalized force

F_x, F_y, F_z - thrust forces in the x, y, or z direction

g - gravity

h_i - normalized magnitude of the ith mode shape

h_0 - magnitude of the plunge mode shape

$I_{xx}, I_{xy}, I_{xz}, I_{yy}, I_{yz}, I_{zz}$ - mass moment or product of inertia

k - reduced frequency ($K = b\omega/V$)

L - Lagrangian

L - aerodynamic moment about the x axis

M - aerodynamic moment about the y axis

$\mathcal{M}$ - mass

M_* - $\partial M/\partial *$

m_j - generalized mass of the jth control surface

N - aerodynamic moment about the z axis

N_i - numerator coefficient of the Pade polynomial

p - roll rate

Q_{ij} - force on the jth mode due to the ith motion

q - pitch rate

$\bar{q}$ - dynamic pressure

r - yaw rate

S - reference area

$\mathcal{S}$ - surface area

s - Laplace variable

T - kinetic energy

V - potential energy

V - velocity

$\mathcal{V}$ - volume

α - angle of attack

ΔP_j - pressure on the jth mode

δ_j - jth control surface

δ_{jo} - magnitude of the jth control mode

φ - bank angle

φ_i - mode shape of the ithe mode of the elastic motion

γ - twist position of wing

η_i - ith generalized coordinate

μ_i - ith generalized mass

ψ - yaw angle

ρ - density

ω - frequency

ω_i - frequency of vibration of the ith mode

ξ_i - motion of the ith mode of the elastic motion

ξ_{io} - magnitude of the ith mode shape

θ - pitch angle

θ_o - magnitude of the pitch mode shape

Vectors

$\underline{a}, \underline{b}, \underline{c}$ - general vectors

$\underline{F}$ - vector of generalized forces producing elastic motion

$\underline{F}_R$ - vector of forces producing rigid-body translation

$\underline{H}$ - angular momentum

$\underline{M}_o$ - vector of moments producing rigid-body rotation

$\underline{P}$ - linear momentum

$\underline{Q}$ - vector of generalized forces

$\underline{q}$ - generalized coordinates

$\underline{R}_o$ - position vector of the origin of the body axis with respect to the inertial frame

$\dot{\underline{R}}_o{}^I$ - velocity of the body axis origin with respect to the inertial frame as seen by an observer in the inertial frame

$\underline{R}_p$ - position vector of point P on the body with respect to the inertial frame

$\dot{\underline{R}}_p{}^I$ - velocity of point P with respect to the inertial frame as seen by an observer in the inertial frame

$\underline{r}_o$ - position vector of the undeformed point P with respect to the body axis

$\underline{u}$ - m vector of inputs to the aircraft

$\underline{V}_o$ - velocity vector of the body axis origin with respect to the inertial frame

$\underline{x}$ - n vector of states of the aircraft

$\underline{y}$ - p vector of the outputs of the aircraft

$\underline{\delta}$ - elastic deformation of point P with respect to the undeformed position

$\underline{\eta}$ - vector of generalized coordinates

$\underline{\Omega}^{B/I}$ - angular velocity vector of the body axis with respect to the inertial frame

Matrices

[A] - n by n plant matrix

$[\tilde{a}]$ - skew symmetric matrix

$[a_i]$ - matrix coefficients of Pade fit equations of motion

[B] - n by m input matrix

[C] - p by n output matrix

[C] - generalized damping matrix

[I] - identity matrix

$[I_C]$ - inertial coupling matrix

$[I_E]$ - elastic inertial matrix

$[I_R]$ - rigid inertial matrix

[K] - generalized stiffness matrix

[M] - generalized mass matrix

[Q(k)] - genralized aerodynamic force matrix

[X] - aircraft mass matrix (including aerodynamics)

[Y] - aircraft damping matrix (including aerodynamics)

[Z] - aircraft stiffness matrix (including aerodynamics)

Abstract

The purpose of this effort was to provide a method of developing a linear model of an elastic aircraft. The model provides the capability to analyze the coupling between the rigid and elastic motion of the aircraft. The method developed in this effort obtains stability derivatives directly from unsteady aerodynamic forces. This results in a state-space model whose states are just the normal aircraft states and rates, the structural coordinates and rates, and the control surface positions and rates. Using a representation of the YF-17 wind tunnel flutter model, it was demonstrated that the methodology developed predicted the required dynamics to make this a viable method of modelling rigid-body and flutter behavior of the model. Flutter control laws were designed for motion about an equilibrium condition represented by a velocity 20% above the flutter velocity. Both classical and modern techniques yielded acceptable control laws. The control laws were also analyzed at off design conditions to check robustness.

MODELLING OF RIGID-BODY AND ELASTIC AIRCRAFT DYNAMICS FOR FLIGHT CONTROL DEVELOPMENT

I. Introduction

With the advent of high gain control systems on high performance, structurally efficient aircraft, the interaction between aerodynamics, structural dynamics and the flight control system, has become a major concern to aircraft designers. This interaction has been termed aeroservoelasticity (ASE). Virtually all major U.S. fighter aircraft in use today have encountered this phenomenon. The F-15 has encountered several ASE problems that have influenced both ground and flight tests (18:Vol I, 8-22). The F/A-18 also had many ASE encounters, primarily in ground tests (18:Vol II, 205-211). The YF-16 and YF-17 also had ASE encounters (7:482). Even the experimental X-29 had ASE problems, although not in flight. The ASE interactions are not limited to new aircraft. The ASE phenomenon has been encountered on the F-4 (18:Vol I, 3-7; 7:482) and many aircraft that were in test phases (18:Vol II, 226-231; 21:10). This interaction, if severe enough, can limit the performance capabilities of an aircraft. Avoidance of such problems calls for the development of an accurate aircraft modelling technique, which can model the interactions correctly.

Background

Typically, the three phenomenon involved in ASE, have been analyzed using separate models for each. The aerodynamic models are usually based on the assumption that the aircraft is a rigid-body. An accurate representation of the aerodynamic shape of the aircraft is required to obtain accurate aerodynamic parameters. The flight control models are typically linearized rigid-body models, using stability derivatives developed for a rigid aircraft. These linear flight control models are then represented by state-space models or through transfer functions. The usual way to account for flexibility is to add an elastic correction factor to the stability derivatives, which does not account for the dynamics of the structure. The structural dynamics model, on the other hand, is concerned mainly in a correct representation of the elastic structure to predict such things as flutter and divergence. Typically, each of the three models leads to an analysis which is never communicated to the other two. This lack of a unified model resulted in the problems that were noted previously.

Correcting the unwanted ASE interactions after the aircraft is built and tested is the most common procedure for dealing with ASE problems. The typical "quick fixes" usually involve the control system. Such fixes have included, adding filters, reducing control system gains, relocating sensors (18; 21:8), introducing flutter placards, and in the case of the X-29, limiting the flight envelope until a better control system could be designed. The process of separate modelling, and then fixing the problems after the aircraft is built limits the potential of modern aircraft design. A unified approach to aircraft modelling can be accomplished, with all three of the phenomenon involved in ASE being accounted for.

The history of developing a unified model for aircraft design goes back well over twenty years. Bisplinghoff and Ashley (1:450-486) and Milne (13) were some of the first to develop the equations of motion for an elastic aircraft. The implementation of these equations was beyond the capabilities of computers of that day. In 1974, the first computer program that addressed the ASE phenomenon was FLEXSTAB (5). FLEXSTAB, developed by Boeing, in cooperation with NASA and the Flight Dynamics Laboratory (FDL), combined aerodynamic, elastic structure and control system models into one computer program for analysis and design purposes

(5). FLEXSTAB was a large and cumbersome program, and as new design and analysis techniques were developed, they could not be easily incorporated into FLEXSTAB. The desire to develop a unified model was continued throughout the 70's. Etkin (6), Schwanz (22; 23), Taylor and Woodcock (26), Warren (29), Rodden (19), NASA engineers (18), and many others (4; 11; 25) all proposed approaches to develop ASE models. However, it became apparent that even by 1984 there was no consensus on the best way to develop a unified model (18).

Recently, the Flight Dynamics Laboratory has developed an in-house computer program called ADAM (Analog and Digital Aeroservoelastic Method). It has the capability to combine unsteady aerodynamics, multi-input multi-output control systems and a structural dynamics model into one analysis package (14:1).

ADAM does have some limitations. The first limitation is that typical stability derivatives that flight controls engineers use are not directly available (14:8). A second difficulty is that ADAM does not include control surface states and thus cannot predict control surface instabilities (2: Vol 1, 12). In addition, ADAM has numerical difficulties inherent in the complexity of the modelling process (14:8). A method is needed to solve these problems in ADAM to make it a true ASE analysis tool.

Purpose

The purpose of this thesis is to develop a method for use in concert with ADAM, in the analysis of ASE problems. This method includes prediction of stability derivatives, addition control surface dynamics, and reducing model complexity.

Scope

This effort will be limited to the development of a linear time-invariant state-space model of an elastic aircraft. This effort will examine only longitudinal motion, although the theory can be easily extended to lateral directional motion. The work is also limited to continuous time models and control systems.

Approach

This thesis will develop the equations of motion of an elastic aircraft, using Lagrange's equations. The resulting equations will then be linearized about an equilibrium point, and the associated non-linear aerodynamics will also be linearized. The kinematic coupling between the rigid-body and elastic motion will be eliminated. Thus the only coupling between the rigid-body and elastic motion will be through the aerodynamics. The resulting second order equations will then be transformed into a first order state-space model for the elastic aircraft. A computer program will then be developed which will automate this procedure. The methodology will be demonstrated with a representation of the YF-17 wind tunnel flutter model.

II. Theoretical Development

It is desired to develop the equations of motion of an elastic aircraft, in the form of Eqs 1 and 2

$$\dot{\underline{x}} = [A]\underline{x} + [B]\underline{u} \quad (1)$$

$$\underline{y} = [C]\underline{x} \quad (2)$$

where

$\underline{x}$	-	n vector of states of the aircraft
$\underline{u}$	-	m vector of inputs to the aircraft
$\underline{y}$	-	p vector of outputs of the aircraft
[A]	-	n by n plant matrix
[B]	-	n by m input matrix
[C]	-	p by n output matrix

There are three general ways that the equations of motion for an elastic aircraft can be determined. The first way, and probably the most common way is to use classical Newtonian dynamics. Etkin (6:122-145) developed the equations of motion for a rigid aircraft using Newtonian dynamics. Milne (13:4), Taylor (26:22), and Bisplinghoff and Ashley (1:450-468) developed the equations of motion for an elastic aircraft using this method. The second method is to use Hamilton's equations to derive the equations of motion (3:1684-1687). The final method of deriving the equations of motion uses Lagrange's equations, as Schwanz had done (22). This is the method used in this development, since it is the most direct and easy to use. Lagrange's equations contain all the information needed to derive the equations of motion.

Lagrange's Equations

Lagrange's equations are based on energy principles. If the aircraft's total potential and kinetic energy can be described, in reference to an inertial frame, then the equations of motion of that aircraft can be derived. Stated in a general form, Lagrange's equations for a holonomic system are

$$d/dt(\partial L/\partial \dot{\underline{\eta}})-\partial L/\partial \underline{\eta}=\underline{Q} \tag{3}$$

$$L=T-V \tag{4}$$

where

T - total kinetic energy,
V - total potential energy,
$\underline{\eta}$ - generalized coordinates,
$\underline{Q}$ - generalized forces, and
$\partial L/\partial \underline{\eta}=\{\partial L/\partial \eta_1 \;\; . \;\; . \;\; . \;\; \partial L/\partial \eta_n\}^T$

Therefore in order to use Lagrange's equations, the kinetic and potential energy of the elastic aircraft must be found.

Consider an elastic body, as shown in Figure 1. The inertial axis origin, O_I, is a fixed point attached to the earth. It will be assumed that the earth is flat and non-rotating. In general the flat non-rotating earth

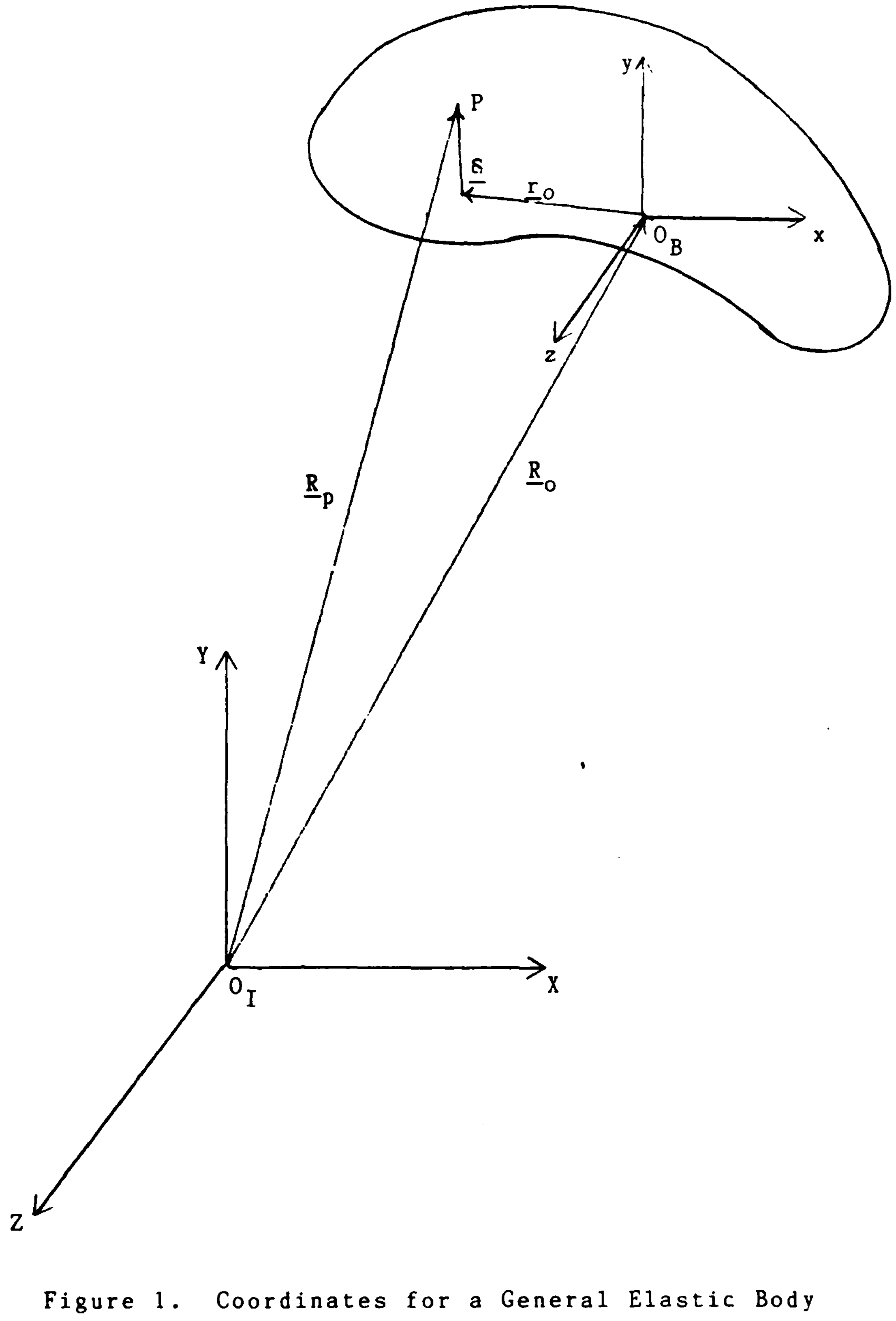


Figure 1. Coordinates for a General Elastic Body

assumption does not have to be made, however, it is reasonable to do so for aircraft motion in subsonic and low supersonic flight (6:148). This assumption will allow the notation to be kept simple, and simplifies the dynamics. The position vector from this inertial origin to any arbitrary point P on the body, $\underline{R}_p$ is given by:

$$\underline{R}_p=\underline{R}_0+\underline{r}_0+\underline{\delta} \tag{5}$$

where

$\underline{R}_0$ - position vector of the body frame origin to the inertial frame origin,
$\underline{r}_0$ - position vector of the undeformed position of P to the body axis origin, and
$\underline{\delta}$ - elastic deformation position vector of point P from the undeformed position.

From this the kinetic and potential energies can be obtained.

<u>Kinetic</u> <u>Energy.</u> The kinetic energy of the body is defined as

$$T=1/2\int_V \rho \dot{\underline{R}}_p{}^I \cdot \dot{\underline{R}}_p{}^I dV \tag{6}$$

where
ρ - density
V - volume,
$\dot{\underline{R}}_p{}^I$ - velocity of point P with respect to O_I as seen by an observer in the inertial reference frame

$\dot{\underline{R}}_p{}^I$ can be represented by

$$\underline{\dot{R}}_p{}^I = \underline{\dot{R}}_o{}^I + d^I/dt(\underline{r}_o + \underline{\delta}) \qquad (7)$$

which can be expanded

$$\underline{\dot{R}}_p{}^I = \underline{\dot{R}}_o{}^I + \underline{\dot{\delta}}^B + \underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta}) \qquad (8)$$

where

$\underline{\Omega}^{B/I}$ - angular velocity of the body axis with respect to the inertial axis

$\underline{\dot{\delta}}^B$- velocity of the deformed point P with respect to the body axis frame

Substituting this back into Eq 6 yields

$$T = 1/2\int_V \rho\{\underline{V}_o + \underline{\dot{\delta}}^B + \underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} . \{\underline{V}_o + \underline{\dot{\delta}}^B + \underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} dV \qquad (9)$$

where, $\underline{\dot{R}}_o{}^I = \underline{V}_o$. Expanding Eq 9 term by term gives

$$\begin{aligned} T = {} & 1/2\int_V \rho \underline{V}_o \cdot \underline{V}_o dV + 1/2\int_V \rho \underline{V}_o \cdot \underline{\dot{\delta}}^B dV + 1/2\int_V \rho \underline{V}_o \cdot \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} dV \\ & + 1/2\int_V \rho \underline{\dot{\delta}}^B . \underline{V}_o dV + 1/2\int_V \rho \underline{\dot{\delta}}^B . \underline{\dot{\delta}}^B dV + 1/2\int_V \rho \underline{\dot{\delta}}^B . \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} dV \\ & + 1/2\int_V \rho \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} . \underline{V}_o dV + 1/2\int_V \rho \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} . \underline{\dot{\delta}}^B dV \\ & + 1/2\int_V \rho \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} . \{\underline{\Omega}^{B/I} x (\underline{r}_o + \underline{\delta})\} dV \end{aligned} \qquad (10)$$

The total mass of the aircraft is defined as

$$M=\int_V \rho \, dV \tag{11}$$

Combining term in Eq 10, and expanding the remaining products yields

$$\begin{aligned} T= & 1/2M\underline{V}_0\cdot\underline{V}_0+\int_V\rho\underline{V}_0\cdot\underline{\dot{\delta}}^B dV+\int_V\rho\underline{V}_0\cdot(\underline{\Omega}^{B/I}x\underline{r}_0)dV+\int_V\rho\underline{V}_0\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV \\ & +1/2\int_V\rho\underline{\dot{\delta}}^B\cdot\underline{\dot{\delta}}^B dV+\int_V\rho\underline{\Omega}^{B/I}\cdot(\underline{r}_0x\underline{\dot{\delta}}^B)dV+\int_V\rho\underline{\Omega}^{B/I}\cdot(\underline{\delta}x\underline{\dot{\delta}}^B)dV \\ & +1/2\int_V\rho(\underline{\Omega}^{B/I}x\underline{r}_0)\cdot(\underline{\Omega}^{B/I}x\underline{r}_0)dV+\int_V\rho(\underline{\Omega}^{B/I}x\underline{r}_0)\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV \\ & +\int_V\rho(\underline{\Omega}^{B/I}x\underline{\delta})\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV \end{aligned} \tag{12}$$

Since $\underline{V}_0$ and $\underline{\Omega}^{B/I}$ do not depend on the position within the volume, they can be pulled outside the integrals. Now the following assumptions are made to help simplify Eq 12. First, it is assumed that the body axis origin is at the center of mass of the equilibrium configuration of the aircraft. This implies that

$$0=\int_V\rho\underline{r}_0 dV \tag{13}$$

The second major assumption is that the aircraft body axis system is the mean axis system, which Milne states as the axis ". . . at every point, the linear and angular momentum of the relative motion with respect to the body axis is identically zero" (13:5). This results in the following

$$0=\int_V \rho\underline{\delta}\,dV=\int_V \rho\underline{r}_0 x\underline{\delta}\,dV \int_V \rho\underline{\dot{\delta}}^B dV=\int_V \rho\underline{r}_0 x\underline{\dot{\delta}}^B dV \tag{14}$$

This reduces the coupling between the overall motion and the elastic deformation (13:27), thus reducing the kinetic energy equation (Eq 12) to

$$T=1/2M\underline{V}_0\cdot\underline{V}_0+1/2\int_V \rho\underline{\dot{\delta}}^B\cdot\underline{\dot{\delta}}^B dV+\int_V \rho\underline{V}_0\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV$$

$$+1/2\int_V \rho(\underline{\Omega}^{B/I}x\underline{r}_0)\cdot(\underline{\Omega}^{B/I}x\underline{r}_0)dV+\int_V \rho(\underline{\Omega}^{B/I}x\underline{r}_0)\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV$$

$$+\int_V \rho(\underline{\Omega}^{B/I}x\underline{\delta})\cdot(\underline{\Omega}^{B/I}x\underline{\delta})dV \tag{15}$$

Noting that the components of the cross product of two vectors expressed in the same orthogonal reference frame, say $\underline{a}x\underline{b}=\underline{c}$, may be obtained from

$$[\tilde{a}]\underline{b}=\underline{c} \tag{16}$$

where $[\tilde{a}]$ is a skew symmetric matrix, defined by

$$[\tilde{a}]=\begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

Eq 15 now becomes

$$T = 1/2 M\underline{V}_0 \cdot \underline{V}_0 + 1/2 \int_V \rho \underline{\dot{\delta}}^B \cdot \underline{\dot{\delta}}^B dV + 1/2 \underline{\Omega}^{B/I} \cdot [I_R] \cdot \underline{\Omega}^{B/I}$$

$$+ \int_V \rho \underline{\Omega}^{B/I} \cdot (\underline{\delta} x \underline{\dot{\delta}}^B) dV + \underline{\Omega}^{B/I} \cdot [I_C] \cdot \underline{\Omega}^{B/I}$$

$$+ 1/2 \underline{\Omega}^{B/I} \cdot [I_E] \cdot \underline{\Omega}^{B/I} \qquad (17)$$

where

$[I_R]$	-	rigid body inertia matrix
$[I_C]$	-	inertial coupling matrix between rigid and elastic motion
$[I_E]$	-	elastic motion inertia matrix

It is at this point that knowledge about the equilibrium conditions is necessary to simplify Eq 17. Since it is desired to develop a linear model, perturbation equations about an equilibrium condition will be developed. The equilibrium condition in this development will be straight and level flight of an *undeformed* airframe, thus the equilibrium body rotation rates (p_e, q_e, r_e) are zero. Thus, $\underline{\Omega}^{B/I}$ is now just a vector of the perturbation rotation rates. Both the coupling and elastic inertial terms of Eq 17 are functions of δ, a perturbation motion. Thus the last three terms of Eq 17 are third, third, and fourth order respectively of the perturbation motion, and thus will be ignored. Thus, Eq 17 becomes

$$T = 1/2 M\underline{V}_0 \cdot \underline{V}_0 + 1/2 \int_V \rho \underline{\dot{\delta}}^B \cdot \underline{\dot{\delta}}^B dV + 1/2 \underline{\Omega}^{B/I} \cdot [I_R] \cdot \underline{\Omega}^{B/I} \qquad (18)$$

As can be seen, this expression for kinetic energy is much simpler than the original equation (Eq 10). The only term that involves the elastic motion is the middle term in Eq 18. It is at this point that some assumptions about the elastic deformations must be made. In general, the elastic deformations of a body can be described by a set of n coupled second order ordinary differential equations, representing the free vibration of the body. These coupled equations can be decoupled by use of a linear transformation (12:143-144). Therefore, in this development, it will be assumed that a set of uncoupled orthogonal modes are available. This assumption will allow diagonal mass and stiffness matrices to be developed for the equations of motion, thus allowing easier decoupling of the elastic and rigid body equations of motion. The elastic deformations are defined as

$$\underline{\delta}=\sum_{i=1}^{n}\varphi_i(x,y,z)\xi_i(t) \tag{19}$$

where,

$\varphi_i(x,y,z)$ - mode shape of the ith mode, and

$\xi_i(t)$ - time dependent motion of the mode

Since the modes are orthogonal, $\int\varphi_i\varphi_j=0$ for $i\neq j$ (12: 143). Now defining the generalized mass as

$$\mu_i = \int_V \rho \phi_i^2 dV \qquad (20)$$

and substituting Eqs 19 and 20 into Eq 18 yields

$$T = 1/2M\underline{V}_0 \cdot \underline{V}_0 + 1/2\sum_{i=1}^{n}\mu_i\dot{\xi}_i^2 + 1/2\underline{\Omega}^{B/I} \cdot [I_R] \cdot \underline{\Omega}^{B/I} \qquad (21)$$

This is the equation that is required for the kinetic energy.

Potential Energy. The potential energy of an elastic structure can be represented by

$$V = -1/2\sum_{i=1}^{n}\mu_i\xi_i^2\omega_i^2 \qquad (22)$$

where

ω_i^2 - the squared natural frequencies of the various modes.

The same assumption about orthogonality that were made earlier is also made here (12:168). The potential energy due to the Earth's gravitation will be dealt with as an external force acting on the aircraft.

Eqs 21 and 22 can now be substituted into Eq 4, the equation for L, and Lagrange's equations formed, and noting that the rigid-body terms do not appear, gives

$$\sum_{i=1}^{n}(\mu_i\ddot{\xi}_i+\mu_i\omega_i^2\xi_i)=\underline{Q} \tag{23}$$

This can be represented in matrix notation as

$$\begin{bmatrix}\mu_1 & & \bigcirc \\ & \ddots & \\ \bigcirc & & \mu_n\end{bmatrix}\begin{bmatrix}\ddot{\xi}_1\\ \cdot \\ \cdot \\ \ddot{\xi}_n\end{bmatrix}+\begin{bmatrix}\omega_1^2\mu_1 & & \bigcirc \\ & \ddots & \\ \bigcirc & & \omega_n^2\mu_n\end{bmatrix}\begin{bmatrix}\xi_1\\ \cdot \\ \cdot \\ \xi_n\end{bmatrix}=\begin{bmatrix}F_1\\ \cdot \\ \cdot \\ F_n\end{bmatrix} \tag{24}$$

where

μ_i - ith generalized mass

ω_i - ith modal frequency

ξ_i - ith generalized modal coordinate

F_i - ith generalized force

In this formulation, the control surface motions will be treated as extra degrees of freedom in the elastic motion. In order to correctly develop the remaining equations for rigid body motion, the definitions of linear and angular momentum must be used. The linear momentum may be found from

$$\underline{P}=\partial T/\partial\underline{V}_o \tag{25}$$

Then the equations of motion for rigid translation become

$$\begin{aligned}\underline{F}_R&=d^I/dt(\underline{P})=d^I/dt(\partial T/\partial\underline{V}_o)\\ &=d^B/dt(\partial T/\partial\underline{V}_o)+\underline{\Omega}^{B/I}x(\partial T/\partial\underline{V}_o)\end{aligned} \tag{26}$$

Using the typical formulation, the aircraft equations of motion become

$$M\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix}+M\begin{bmatrix}0 & -r & q\\ r & 0 & -p\\ -q & p & 0\end{bmatrix}\begin{bmatrix}u\\ v\\ w\end{bmatrix}=\begin{bmatrix}X+F_x-Mg\sin\theta\\ Y+F_y-Mg\sin\varphi\cos\theta\\ Z+F_z-Mg\cos\varphi\cos\theta\end{bmatrix} \tag{27}$$

where

M - mass
u - velocity in the x direction
v - velocity in the y direction
w - velocity in the z direction
p - roll rate
q - pitch rate
r - yaw rate
X - aerodynamic force in the x direction
Y - aerodynamic force in the y direction
Z - aerodynamic forec in the z direction
F_x - thrust in the x direction

F_y - thrust in the y direction

F_z - thrust in the z direction

g - acceleration of gravity
θ - pitch angle
φ - bank angle

with the following equations for kinematics and trajectory

$$\begin{bmatrix}p\\ q\\ r\end{bmatrix}=\begin{bmatrix}1 & 0 & -\sin\theta\\ 0 & \cos\varphi & \sin\varphi\cos\theta\\ 0 & -\sin\varphi & \cos\varphi\cos\theta\end{bmatrix}\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} \tag{28}$$

$$\begin{bmatrix}u\\ v\\ w\end{bmatrix}=\begin{bmatrix}c\theta c\psi & c\theta s\psi & -s\theta\\ s\varphi s\theta c\psi-c\varphi s\psi & s\varphi s\theta s\psi+c\varphi c\psi & s\varphi c\theta\\ c\varphi s\theta c\psi-s\varphi s\psi & c\varphi s\theta s\psi-s\varphi c\psi & c\varphi c\theta\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} \tag{29}$$

where

c - cosine
s - sine
ψ - yaw angle

A similar method can be used to develop the equations of motion for the attitude motion. The angular momentum is given by

$$\underline{H}=\partial T/\partial \underline{\Omega}^{B/I} \tag{30}$$

The equations for rigid body rotation become

$$\begin{aligned}\underline{M}_0 &= d^I/dt(\underline{H})=d^I/dt(\partial T/\partial \underline{\Omega}^{B/I}) \\ &= d^B/dt(\partial T/\partial \underline{\Omega}^{B/I})+\underline{\Omega}^{B/I}x(\partial T/\partial \underline{\Omega}^{B/I})\end{aligned} \tag{31}$$

These equations (Eqs 26 and 31) are sometimes termed Lagrange's equations in quasi- coordinates. Eq 31 in matrix notation becomes

$$\begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz}\end{bmatrix}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} + \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0\end{bmatrix}\begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & II_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz}\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \quad \begin{bmatrix}L\\ M\\ N\end{bmatrix} \tag{32}$$

where

$I_{xx}, I_{xy}, I_{xz}, I_{yy}, I_{yz}, I_{zz}$ - mass moment or product of inertia
L - aerodynamic moment about the x axis
M - aerodynamic moment about the y axis
N - aerodynamic moment about the z axis

Eqs 24, 27, 28, 29, and 32 comprise the desired equations of motion. The forces in Eqs 24, 27, and 32 are now function of both the rigid and elastic motion.

Longitudinal Equations of Motion

The desired equations of motion, Eqs 1 and 2, are linear in nature. The above equations (Eqs 27, 28, 29, and 32) still contain non-linear terms which need to be further linearized. In this development, however, only the longitudinal equations will be developed. Therefore, only those equations appropriate to the longitudinal motion will be addressed. The equilibrium condition is again straight and level flight, but now using longitudinal motion, equations reduce to the following scalar equations

$$\mathcal{M}\dot{u}+\mathcal{M}qw=-X+F_x-\mathcal{M}g\sin\theta \tag{33}$$

$$\mathcal{M}\dot{w}-\mathcal{M}qu=-Z+F_z-\mathcal{M}g\cos\theta \tag{34}$$

$$I_{yy}\dot{q}=M \tag{35}$$

$$q=\dot{\theta} \tag{36}$$

$$u=\dot{x}\cos\theta-\dot{z}\sin\theta \tag{37}$$

$$w=\dot{x}\sin\theta+\dot{z}\cos\theta \tag{38}$$

The attitude equations are uncoupled from the trajectory equations (Eqs 37 and 38), and thus the trajectory equations will be disregarded in this development. Now it will be assumed that each variable will be the sum of its' equilibrium value and perturbed value (i.e. $u=u_e+u'$), and the derivatives of the equilibrium values are zero. It will also be assumed that the mass and inertia remain essentially constant. If the thrust is to be used as an input it can be used as a control input. Substituting these into Eqs 33-38 yields

$$M\dot{u}'+M(q')(w_e+w')=-(X_e+X')+F_x-Mg\sin(\theta_e+\theta') \tag{39}$$

$$M\dot{w}'-M(q')(u_e+u')=-(Z_e+Z')+F_z-Mg\cos(\theta_e+\theta') \tag{40}$$

$$I_{yy}\dot{q}'=M_e+M' \tag{41}$$

$$(q')=\dot{\theta} \tag{42}$$

$$u'=\dot{x}'\cos(\theta_e+\theta')-\dot{z}'\sin(\theta_e+\theta') \tag{43}$$

$$u'=\dot{x}'\sin(\theta_e+\theta')+\dot{z}'\cos(\theta_e+\theta') \tag{44}$$

Now taking all the terms that strictly deal with equilibrium conditions yields Eqs 45-47 while the remaining terms result in the perturbation equations, Eqs 48-53.

$$0=-X_e+F_x-Mg\sin\theta_e \tag{45}$$

$$0=-Z_e+F_z-Mg\cos\theta_e \tag{46}$$

$$M_e=0 \tag{47}$$

$$M\dot{u}+M(w_e q'+w'q')=-X'-Mg\sin\theta' \tag{48}$$

$$M\dot{w}+M(u_e q'+u'q')=-Z' \tag{49}$$

$$I_{yy}\dot{q}'=M' \tag{50}$$

$$q'=\dot{\theta}' \tag{51}$$

$$u'=\dot{x}' \tag{52}$$

$$w'=\dot{z}' \tag{53}$$

The equilibrium equations and the perturbation equations are identical to those developed in Taylor (26:39-44). Assuming that the terms where products of perturbation quantities exists are small results in the linear set of equations in

Eq 54, where the primed superscript is dropped.

$$\begin{bmatrix} M & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & I_{yy} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0 & 0 & Mw_e \\ 0 & 0 & Mu_e \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \end{bmatrix} = \begin{bmatrix} -X-Mg\theta \\ -Z \\ M \end{bmatrix} \qquad (54)$$

The equations can be combined with the elastic equations of motion to yield the following matrix equation:

$$\left[\begin{array}{ccc|ccc} M & 0 & 0 & & & \\ 0 & M & 0 & & \bigcirc & \\ 0 & 0 & I_{yy} & & & \\ & & & \mu_1 & & \\ & \bigcirc & & & \ddots & \\ & & & & & \mu_n \end{array}\right] \begin{bmatrix} \ddot{x} \\ \ddot{z} \\ \ddot{\theta} \\ \ddot{\epsilon}_1 \\ \vdots \\ \ddot{\epsilon}_n \end{bmatrix} + \left[\begin{array}{ccc|c} 0 & 0 & Mw_e & \\ 0 & 0 & Mu_e & \bigcirc \\ 0 & 0 & 0 & \\ & & & \\ & \bigcirc & & \\ & & & \end{array}\right] \begin{bmatrix} \dot{x} \\ \dot{z} \\ \dot{\theta} \\ \dot{\epsilon}_1 \\ \vdots \\ \dot{\epsilon}_n \end{bmatrix}$$

$$+ \left[\begin{array}{ccc|ccc} 0 & 0 & Mg & & & \\ 0 & 0 & 0 & & \bigcirc & \\ 0 & 0 & 0 & & & \\ & & & \omega_1{}^2\mu_1 & & \\ & \bigcirc & & & \ddots & \\ & & & & & \omega_n{}^2\mu_n \end{array}\right] \begin{bmatrix} x \\ z \\ \theta \\ \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix} = \begin{bmatrix} -X \\ -Z \\ M \\ F_1 \\ \vdots \\ F_n \end{bmatrix} \qquad (55)$$

These equations are identical to those of Schwanz (23:54). The linear aerodynamics which drive these equations must now be determined to complete the model of the flexible aircraft. From this point on, only the short period motion will be considered. Thus the first equation in Eq 55 will not be considered and there will not be an x state.

Linear Aerodynamics

Since the equations of motion of the aircraft have been linearized, the aerodynamics associated with this model must also be linearized. Etkin (6:159) and Taylor (26:45) have both shown that the aerodynamics can be approximated linearly. Ignoring symmetric and antisymmetric motion interactions, and neglecting most higher order aerodynamic derivatives, the longitudinal aerodynamic forces are of the following form

$$M=M_u u'+M_q q'+M_\alpha \alpha+M_{\dot\alpha}\dot\alpha+\sum_{j=1}^{m}(M_{\delta_j}\delta_j+M_{\dot\delta_j}\dot\delta_j+M_{\ddot\delta_j}\ddot\delta_j)$$

$$+\sum_{i=1}^{n}(M_{\xi_i}\xi_i+M_{\dot\xi_i}\dot\xi_i+M_{\ddot\xi_i}\ddot\xi_i) \qquad (56)$$

where,

$M_u=\partial M/\partial u$, etc.

In general these derivatives are not directly available from the unsteady aerodynamic methods such as the Doublet Lattice (8), which is used in flutter prediction work. Rodden and Giesing (20) have shown however, that stability derivatives can be obtained from such methods. In developing unsteady aerodynamic forces and moments, it is assumed that the motion is purely oscillatory and of the form $e^{i\omega t}$. The resulting generalized forces therefore contain real and imaginary parts. The aerodynamics are determined at a constant altitude and Mach number for various reduced frequencies ($K=\omega b/V$, where b is a reference semi-chord and V is the velocity). This results in a set of forces for each degree of freedom for each reduced frequency.

One way to approximate the frequency dependent is to use a Pade fit as a function of reduced frequency (14:3-4). In general the equations of motion can be represented by the second order equations

$$[M]\ddot{\underline{q}}+[C]\dot{\underline{q}}+[K]\underline{q}=-(\rho V^2 S/2)[Q(K)]\underline{q} \qquad (57)$$

where

[M] - generalized mass matrix
[C] - generalized damping matrix
[K] - generalized stiffness matrix
[Q(K)] - generalized aerodynamic matrix
$\underline{q}$ - generalized coordinate vector
V - velocity
S - reference area

The [Q(K)] matrix is a matrix of complex coefficients based on simple harmonic motion. In order to solve the flutter problem in a classical way, many different values of K and the associated [Q(K)] matrices are required. An alternate technique is to curve fit each coefficient in the [Q(K)] matrix with respect to K, using Pade polynomials. The polynomials are of the form

$$Q_{ij}(k)=\{N_0+N_1(ik)+N_2(ik)^2+N_3(ik)^3+N_4(ik)^4\}/$$
$$\{1+N_0+D_1(ik)+D_2(ik)^2\} \qquad (58)$$

for a fourth order over second order fit. Then substituting bs/V=ik into Eq 58 yields

$$Q_{ij}(s)=\{N_0+N_1(bs/V)+N_2(bs/V)^2+N_3(bs/V)^3+N_4(bs/V)^4\}/$$
$$\{1+N_0+D_1(bs/V)+D_2(bs/V)^2\} \qquad (59)$$

Taking the Laplace transform of the right side of Eq 57, and equating terms of like power in s, results in the following equation

$$\{[I]s^4+[a_3]s^3+[a_2]s^2+[a_1]s+[a_0]\}\underline{g}(s)=0 \qquad (60)$$

As can be seen, this method, increases the order of the system by the multiple associated with the order of the denominator of the Pade fit. The system order is therefore altered unless the Pade denominator is a constant. This makes it extremely difficult to form a state-space model of the aircraft in which all the states relate directly to physical quantities. Another method, suggested by Rodden and Giesing (20) allows the state variables to be explicitly stated, and may also reduce the order of the model. This is the method that will be used in this study.

In order to derive the equations necessary to find the derivatives some assumptions must be made. In the analysis of the unsteady aerodynamic forces, the rigid-body motions are assumed to be pitch (q) and plunge (h). In aircraft dynamics, plunge is typically replaced with angle of attack (α). Plunge is pure vertical motion, and is related directly to w by ($\dot{h}$=w) if measured from the center of mass. If α is small, which can be assumed since perturbation motion is being discussed, then; $\alpha = w/V = \dot{h}/V$. And since the equilibrium condition assumes level flight then $\alpha = \theta$ for the equilibrium conditions.

The method of Rodden and Giesing was developed to calculate dynamic stability derivatives from unsteady aerodynamics (20). This methodology is summarized here, using the moment equation as an example. First, it is assumed that stability derivatives can be represented by a Maclaurian series

$$M = M_0 + M_\alpha \alpha + M_{\dot{\alpha}} \dot{\alpha} + M_{\ddot{\alpha}} \ddot{\alpha} + M_q q + \ . \ . \ . \qquad (61)$$

Oscillatory theory cannot predict the M_0 term, and therefore it will be omitted. Next it is assumed that the pitching and plunging motion are oscillatory and are of the form $\alpha = \theta = \theta_0 e^{i\omega t}$ and $h = h_0 e^{i\omega t}$, and thus for pitching motion

$$M = \text{Real}(M_e i\omega t) \tag{62}$$

Hence the complex pitching amplitude M due to pitching motion can be represented by the series expansion to first order

$$\bar{M} = \theta_0 [M_\alpha + i\omega(M_{\dot{\alpha}} + M_q)] \tag{63a}$$

Similarly for plunging motion with $\alpha = \dot{h}/V$

$$\bar{M} = h_0/2\bar{c}\,(i\omega M_\alpha - \omega^2 M_{\dot{\alpha}}) \tag{63b}$$

The terms of Eqs 63 can now be directly related to the appropriate terms of the matrix of complex frequency dependent aerodynamic coefficients from the Doublet Lattice method. According to Rodden and Giesing, the steady dynamic stability derivatives are defined by the limiting values as k approaches 0, so in general, a small value of k should be used. For greater detail on the procedure used by Rodden and Giesing, the reader is referred to their paper (20).

The application of the procedure outlined above in this effort is as follows. The unsteady aerodynamic forces were obtained from FASTOP (30) which uses Doublet Lattice aerodynamics. The generalized forces for a reduced frequency are

$$Q_{ij} = (1/s^2)\iint_S (\Delta P_j/\bar{q})(h_i/s)\,dS \qquad (64)$$

where

Q_{ij} - the force on the ith mode due to the jth modal deflection
ΔP_j - pressure on the jth mode
h_i - magnitude of the ith mode
s - reference length
S - surface area
q - dynamic pressure

In order to obtain stability derivatives the Q_{ij} terms must be normalized by both the ith and jth mode shape magnitudes as was demonstrated by Noll with lateral-directional derivatives (15). To develop dimensional stability derivatives the Q_{ij} terms must be multiplied by the dynamic pressure (q), and the reference area (S). Also, any Q_{ij} terms which are related to moments must also be multiplied by a reference length ($\bar{c}$ - reference chord). Thus the terms in Eqs 65-73 are now dimensional quantities. Eqs 65 are the rigid-body results of Rodden and Giesing in dimensional form

<u>Pitch and Plunge Influence on "Rigid Forces"</u>

$$Q_{ZZ} = (i\omega Z_\alpha - \omega^2 Z_{\dot{\alpha}})(h_o/V)(h_o)e^{i\omega t} \qquad (65a)$$

$$Q_{MZ} = (i\omega M_\alpha - \omega^2 M_{\dot{\alpha}})(h_o/V)(\theta_o)e^{i\omega t} \qquad (65b)$$

$$Q_{ZM} = \{Z_\alpha + i\omega(Z_{\dot{\alpha}} + Z_q)\}(\theta_o)(h_o)e^{i\omega t} \qquad (65c)$$

$$Q_{MM} = \{M_\alpha + i\omega(M_{\dot{\alpha}} + M_q)\}(\theta_o)(\theta_o)e^{i\omega t} \qquad (65d)$$

Rodden and Giesing's method for rigid-body motion can easily be extended further to obtain the elastic and control surface influences on the rigid body forces and visa versa, along with the interactions among themselves. These are shown in Eqs 66-73.

<u>Elastic Influences on "Rigid Forces"</u>

$$Q_{Z\xi_i} = (Z_{\xi_i} + i\omega Z_{\dot{\xi}_i} - \omega^2 Z_{\ddot{\xi}_i})(\xi_{i_o})(h_o)e^{i\omega t} \tag{66a}$$

$$Q_{M\xi_i} = (M_{\xi_i} + i\omega M_{\dot{\xi}_i} - \omega^2 M_{\ddot{\xi}_i})(\xi_{i_o})(\theta_o)e^{i\omega t} \tag{66b}$$

<u>Pitch and Plunge Influence on "Elastic Forces"</u>

$$Q_{\xi_i Z} = (i\omega F_{\xi_i\alpha} - \omega^2 F_{\xi_i\dot{\alpha}})(h_o/V)(\xi_{i_o})e^{i\omega t} \tag{67a}$$

$$Q_{\xi_i M} = \{F_{\xi_i\alpha} + i\omega(F_{\xi_i\dot{\alpha}} + F_{\xi_i q})\}(\theta_o)(\xi_{i_o})e^{i\omega t} \tag{67b}$$

<u>Elastic Influence on "Elastic Forces"</u>

$$Q_{\xi_j\xi_i} = (F_{\xi_j\xi_i} + i\omega F_{\xi_j\dot{\xi}_i} - \omega^2 F_{\xi_j\ddot{\xi}_i})(\xi_{i_i})(\xi_{j_j})e^{i\omega t} \tag{68}$$

<u>Control Surface Influence on "Rigid Forces"</u>

$$Q_{Z\delta_i} = (Z_{\delta_i} + i\omega Z_{\dot{\delta}_i} - \omega^2 Z_{\ddot{\delta}_i})(\delta_{i_o})(h_o)e^{i\omega t} \tag{69a}$$

$$Q_{M\delta_i} = (M_{\delta_i} + i\omega M_{\dot{\delta}_i} - \omega^2 M_{\ddot{\delta}_i})(\delta_{i_o})(\theta_o)e^{i\omega t} \tag{69b}$$

<u>Pitch and Plunge Influence on "Control Surface Forces"</u>

$$Q_{\delta_i Z} = (i\omega F_{\delta_i\alpha} - \omega^2 F_{\delta_i\dot{\alpha}})(h_o/V)(\delta_{i_o})e^{i\omega t} \tag{70a}$$

$$Q_{\delta_i M} = \{F_{\delta_i\alpha} + i\omega(F_{\delta_i\dot{\alpha}} + F_{\delta_i q})\}(\theta_o)(\delta_{i_o})e^{i\omega t} \tag{70b}$$

Control Surface Influence on "Elastic Forces"

$$Q_{\xi_j\delta_i} = (F_{\xi_j\delta_i} + i\omega F_{\xi_j\delta_i} - \omega^2 F_{\xi_j\ddot{\delta}_i})(\delta_{i_i})(\xi_{j_j})e^{i\omega t} \qquad (71)$$

Elastic Influence on "Control Surface Forces"

$$Q_{\delta_j\xi_i} = (F_{\delta_j\xi_i} + i\omega F_{\delta_j\dot{\xi}_i} - \omega^2 F_{\delta_j\ddot{\xi}_i})(\xi_{i_i})(\delta_{j_j})e^{i\omega t} \qquad (72)$$

Control Surface Influence on "Control Surface Forces"

$$Q_{\delta_j\delta_i} = (F_{\delta_j\delta_i} + i\omega F_{\delta_j\delta_i} - \omega^2 F_{\delta_j\ddot{\delta}_i})(\delta_{i_i})(\delta_{j_j})e^{i\omega t} \qquad (73)$$

In FASTOP, to find the steady derivatives, k is set to zero, thus only the real terms of the above equations appear. The remaining derivatives are found at a small value of k. The expansion was kept to second order which results in no increase in system order. If higher order dynamics are needed, they can be easily added by adding higher order stability derivatives. This will, of course increase the order of the system.

Resulting Equations

The stability derivatives from Eqs 65-73 can now be moved to the left hand side of Eq 55, thus filling the matrices with terms. The only terms left on the right hand side of Eq 55 are the generalized forces due to the torques applied to the control surface hinges from the control inputs, as shown in Eq 74.

$$[X]\ddot{\underline{x}}+[Y]\dot{\underline{x}}+[Z]\underline{x}=\underline{Q} \quad (74)$$

It is shown Kane (9:81-83) that the generalized forces, $\underline{Q}$, due to the control torques can be represented by

$$Q_i=\sum_{j=1}^{m}\partial\underline{\Omega}/\partial\dot{\eta}_i \cdot T_j \quad (75)$$

where

T_j - torques applied to the control surface hinges

This can be represented by

$$\underline{Q}=[\partial\underline{\Omega}/\partial\dot{\underline{\eta}}_i][T_1 \;.\;.\;.\; T_m]^T \quad (76)$$

The torque can be related to the input signal to the surface by

$$T_j=m_j\delta_j \quad (77)$$

where

m_i - the generalized mass of the ith control surface
$\underline{\delta}^i$ - is the command signal to the ith control surface

Now letting $\underline{\delta}=[\delta_1 \;.\;.\;.\delta_m]$ Eq 75 becomes

$$\underline{Q}=[E]\underline{u} \quad (78)$$

where

$$[E] = [\partial\underline{\Omega}/\partial\dot{\underline{\eta}}_i][\text{diag } m_j]$$

Thus Eq 74 becomes

$$[X]\ddot{\underline{x}} + [Y]\dot{\underline{x}} + [Z]\underline{x} = [E]\underline{u} \tag{79}$$

If the matrix [X] can be inverted, then Eq 79 can be rearranged into the typical state-space representation

$$\begin{bmatrix}\dot{\underline{x}}\\ \ddot{\underline{x}}\end{bmatrix} = \begin{bmatrix}0 & I\\ -X^{-1}Z & -X^{-1}Y\end{bmatrix}\begin{bmatrix}\underline{x}\\ \dot{\underline{x}}\end{bmatrix} + \begin{bmatrix}0\\ -X^{-1}E\end{bmatrix}\underline{u} \tag{80}$$

Eq 80 is the desired equation in the form of Eq 1. Eq 80 and the procedure for obtaining the dimensional stability derivatives from the generalized forces (Eqs 65-73) were programmed into a computer procedure called MAC (Methodology for Aeroelasticity and Controls). Appendix B is the User's Manual, and Appendix C are the subroutines that comprise MAC.

III. YF-17 Model Development

In order to demonstrate the methodology developed in the previous section, a simple model that exhibited flutter characteristics was needed. A mathematical representation of the YF-17 wind tunnel flutter model was used. This model had the required structural properties needed to validate the methodology. Second, only the first two elastic modes were necessary to adequately predict flutter. This results in a simple representation of the model dynamics.

The model used is shown in Figure 2. It is represented by six modes; the rigid body pitch and plunge modes, two structural modes (first wing bending and first wing torsion modes), and two control surface modes (the trailing edge flap and an all moveable horizontal tail). This model will allow full testing of the influences desired. Table I contains the generalized masses and structural frequencies for the model. The model developed includes an elastic wing, connected by a rigid extension to a rigid tail. The mode shapes for the elastic modes were obtained from Noll's work (16). The geometric and structural data were entered into FASTOP, a flutter prediction program (30). FASTOP was used to determine the generalized forces for reduced

TABLE I

YF-17 FLUTTER MODEL STRUCTURAL DATA

Mode	Generalized Mass	Vibration Frequency (Hz)
1	7.061	0.0
2	77.777	0.0
3	0.3115	4.628
4	0.1019	7.186
5	0.1027	50.0
6	0.1	60.0

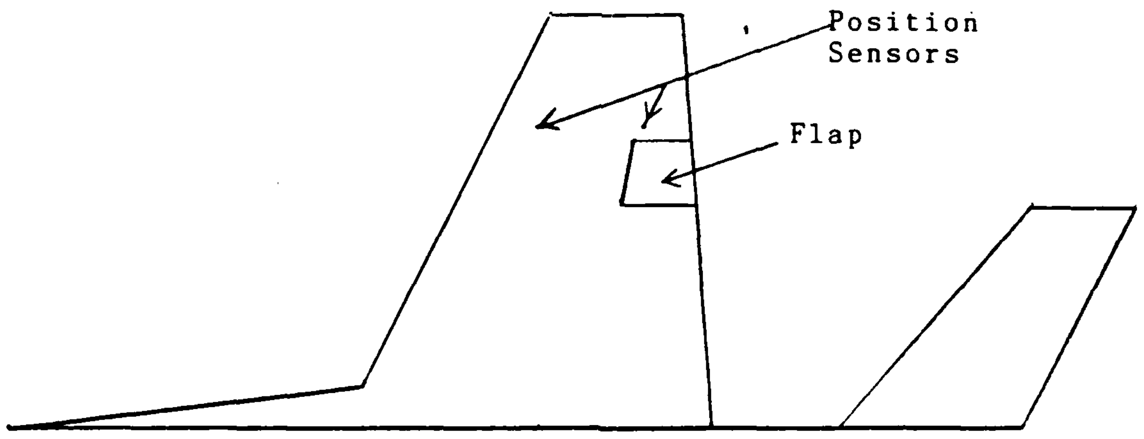


Figure 2. YF-17 Model Planform

frequencies of 0.0 and 0.02. The flutter prediction capability of FASTOP, which uses the P-K method of flutter prediction, was also used to check the validity of this prediction method. The flutter prediction capability of ADAM (2:Vol 1, 24-25), which uses a 'velocity root locus' technique to predict flutter, was also used as a check.

Two sensors were chosen near the trailing edge flap, at 40.154 inches along the span, as shown in Figure 2. Each sensor measures translational motion at that wing location. The two outputs are then differenced and divided by the difference between the two sensors, to obtain the twist angle at that span station. The resulting output matrix, [C] is then a combination of the two structural mode shapes at that span station. Appendix A contains the output matrix used. The sensor and actuator dynamics are not modelled in this development.

IV. Results and Discussion

The resulting equations (Eq 80) are linearized approximations to the actual aircraft dynamics. Where the aerodynamics are linear with respect to the reduced frequencies, this model should successfully capture the essential dynamics and thus predict the low frequency flutter, with the added benefit of predicting rigid body stability derivatives. The resulting state space model will also be of lower order than that obtained using aerodynamics modelled using the Pade approximant method, since it will not contain the often unnecessary higher order aerodynamics associated with curve-fitting the aerodynamics in the frequency domain.

In order to prove that this method is a useful way to model elastic aircraft dynamics for flight control use, three tasks were analyzed. First, the ability of this method to predict stability derivatives, and thus, the ability to predict rigid body dynamics, was investigated. Next, the methodology's ability to predict flutter and associated structural instabilities was investigated. Finally, the model was used to develop a workable flutter control law.

TABLE II

STABILITY DERIVATIVE COMPARISON FOR THE YF-17 MODEL

Derivative	Datcom	MAC
$C_{L\alpha}$	4.305	4.142
$C_{M\alpha}$	-0.6501	-0.5038
$C_{L\dot{\alpha}}$	2.623	2.276
$C_{M\dot{\alpha}}$	-4.699	-7.312
C_{Lq}	6.03	8.145
C_{Mq}	-6.799	-7.474
$C_{L\delta_e}$	1.054	1.153
$C_{M\delta_e}$	-1.886	-1.909
$C_{L\delta_f}$	0.1432	0.233
$C_{M\delta_f}$	-0.0751	-0.4366

Stability Derivative Prediction

The capability of MAC to predict rigid body stability derivatives was validated using the YF-17 model. The stability derivatives from MAC were compared to the analytical methods of Digital Datcom (28). As can be seen in Table II, MAC does an acceptable job in predicting most stability derivatives, with the exception of the flap derivatives. If there are better sources for stability derivatives, MAC has the capability to incorporate them into the analysis.

Flutter Prediction

The capability of MAC to predict flutter was also evaluated with the YF-17 model. The results from MAC were compared to two different flutter prediction methods. The first is FASTOP, which uses the P-K method of predicting flutter speeds and frequencies (30). The second is ADAM, which uses a 'velocity root locus' technique to predict flutter speeds and frequencies (2:Vol 1, 24-25). Figure 3 is a plot of ADAM's root locus. MAC uses a similar technique using Eq 80. Figure 4 shows the plot of the velocity root locus of MAC for various speeds. As can be seen by Figures 3 and 4, both methods are in very close agreement in predicting the dynamics of the structural modes. All three methods agree very well in predicting the flutter speed and frequency (Table III). Using FASTOP as a basis, it can be seen that both ADAM and MAC slightly underpredict the speed of flutter while slightly overpredicting the frequency. All the predictions are within 5% of one another.

In comparing the eigenvalues of MAC with ADAM (Table IV) it is seen that the higher order aerodynamic roots associated with the Pade fit method of ADAM do not seem to be important to the dynamics involved. This is similar to the result that Pasquini reported (17:31-32).

TABLE III

COMPARISON OF FLUTTER PREDICTION METHODS FOR THE YF-17 MODEL

Flutter Prediction Method	Flutter Speed (ft/sec)	Flutter Frequency (Hertz)
FASTOP	387	6.10
ADAM	371	6.25
MAC	382	6.18

TABLE IV

EIGENVALUE COMPARISON OF MAC WITH ADAM

FOR THE 458 FT/SEC CASE

Eigenvalue	MAC	ADAM
Rigid Body	0.000	1.838+3.314i
	0.000	1.838-3.314i
	-2.412+5.586i	-4.215+6.084i
	-2.412-5.586i	-4.215-6.084i
Elastic	-5.890+34.33i	-5.901+34.26i
	-5.890-34.33i	-5.901-34.26i
	3.371+36.67i	3.485+36.66i
	3.371-36.67i	3.485-36.66i
Control Surface	-0.518+314.4i	-
	-0.518-314.4i	-
	-12.93+373.3i	-
	-12.93-373.3i	-
Aerodynamic	-	-93.92+184.7i
	-	-93.92-184.7i
	-	-94.01+184.0i
	-	-94.01-184.0i
	-	-94.34+184.2i
	-	-94.34-184.2i
	-	-94.23+184.3i
	-	-94.23-184.2i

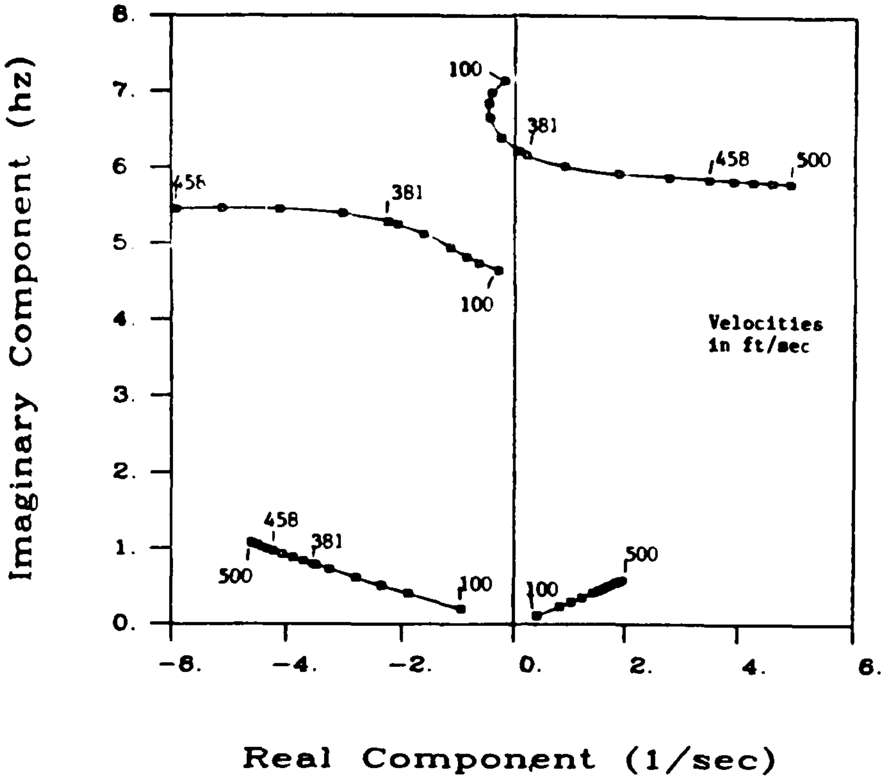


Figure 3. Velocity Root Locus for the Unaugmented YF-17 Model Using ADAM

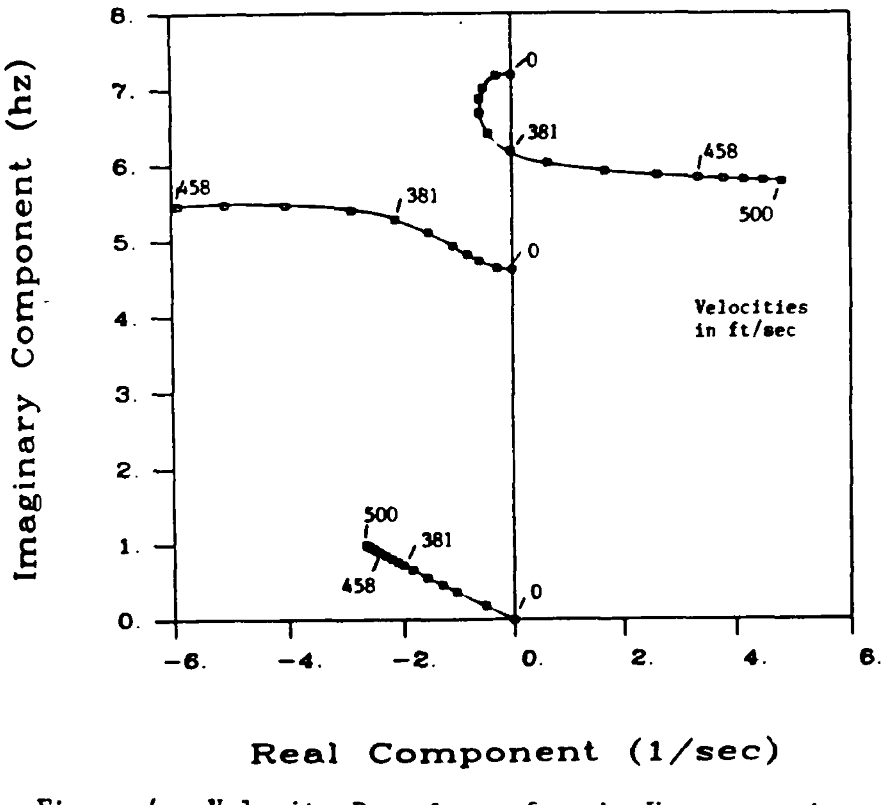


Figure 4. Velocity Root Locus for the Unaugmented YF-17 Model Using MAC

Control Design

From the previous two sections it can be concluded that Eq 80 accurately captures the essential dynamics of the YF-17 model. The model from MAC was then used to develop a flutter control system, using both classical and modern control techniques. The benefit of using this method of developing the aircraft equations of motion is that the state-space model (Eq 80) is formed from the second order form of the equations of motion (Eq 79). The state vector then, is comprised of the generalized coordinates and their rates. In contrast, if the Pade approximation method is used, the state-space model must be formed from the transfer function equation (Eq 60). Typically, this introduces additional states, and yields states which are not the generalized coordinates and their rates.

Both control designs were based on the YF-17 model at 1.2 times the flutter speed (458 ft/sec). All the eigenvalues of this condition are in Table V.

Classical Design. Transfer functions were determined from the state-space model for the above condition using MATRIXx (10). The transfer function from the sensor to the flap (y/δ_f) was then used to develop a feedback control

TABLE V

EIGENVALUES OF THE UNAUGMENTED YF-17 AT 458 FT/SEC

Eigenvalue	Frequency (Hz)
0.000	0.000
0.000	
-2.412+5.586i	0.889
-2.412-5.586i	
-5.890+34.33i	5.464
-5.890-34.33i	
3.371+36.67i	5.836
3.371-36.67i	
-0.518+314.4i	50.04
-0.518-314.4i	
-12.93+373.3i	59.42
-12.93-373.3i	

law. Using root locus techniques in TOTAL (27), the gain was varied until the unstable structural mode became stable while keeping the remaining structural mode and the short period roots stable. The gain chosen was - 50. Figure 5 is the root locus for the γ/δ_e transfer function, with twist position feedback. As can be seen, the unstable structural mode is made stable, however, the very lightly damped control surface mode goes slightly unstable. This is not of much concern since the control surface actually has damping due to frictional forces, which are not accounted for in this model. The resulting roots of this condition with the control system are in Table VI.

This simple control system was then tested at off design conditions. As can be seen from the 'velocity root

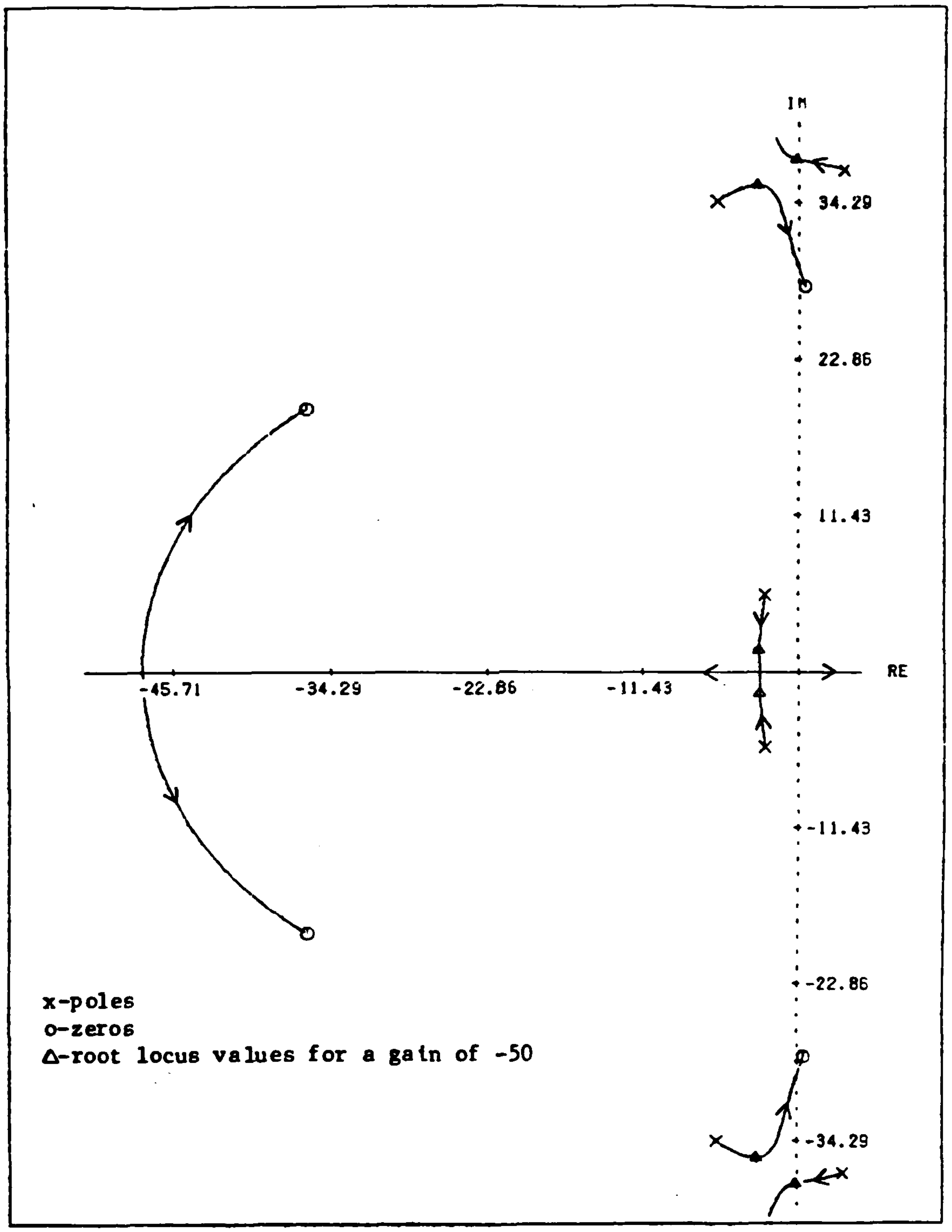


Figure 5. Root Locus of γ/δ_f of the YF-17 Model at 458 ft/sec

TABLE VI

EIGENVALUES OF THE YF-17 MODEL AT 458 FT/SEC

USING TWIST POSITION FEEDBACK

Eigenvalue	Frequency (Hz)
0.000	0.000
0.000	
-2.826+1.579i	0.251
-2.826-1.579i	
-3.003+35.54i	5.556
-3.003-35.54i	
-0.082+37.44i	5.959
-0.082-37.44i	
0.460+312.1i	49.67
-0.460-312.1i	
-12.92+373.3i	59.42
-12.92-373.3i	

locus' in Figure 6, the control system keeps the structural modes and the short period mode stable for all conditions above 250 ft/sec up to the design condition, thus yielding an acceptable control system, if it is implemented after the model reaches 250 ft/sec.

Modern Design. Again using the 1.2 flutter speed condition as the design point, a state-space feedback control law was developed. Eigenvalue assignment was used to assign the eigenvalues as shown in Table VII. The resulting feedback matrix was found using MATRIXx (10).

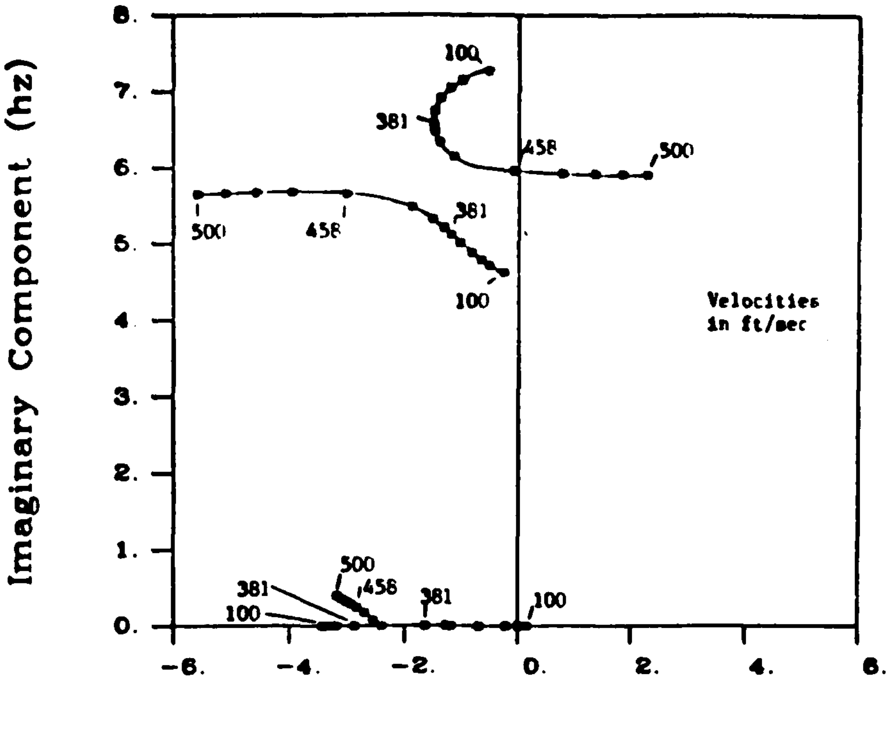


Figure 6. Velocity Root Locus for the YF-17 Model with Twist Position Feedback

TABLE VII

EIGENVALUES OF THE YF-17 MODEL AT 458 FT/SEC USING STATE-SPACE FEEDBACK

Eigenvalue	Frequency (Hz)
0.000	0.000
0.000	
-2.412+5.594i	0.890
-2.412-5.594i	
-5.889+34.33i	5.464
-5.889-34.33i	
-3.375+36.67i	5.836
-3.375-36.67i	
-10.00+314.4i	50.04
-10.00-314.4i	
-12.93+373.3i	59.42
-12.93-373.3i	

The feedback matrix was then used at the off design conditions. This state-space feedback system kept the structural and short period modes stable for all the conditions above 100 ft/sec, up to the design conditions, and even beyond (Figure 7).

Quite often the states are not readily measurable. However, the modelling technique as outlined in this effort can be used in the design of an estimator or observer. This model can be usefully employed in a modern control design.

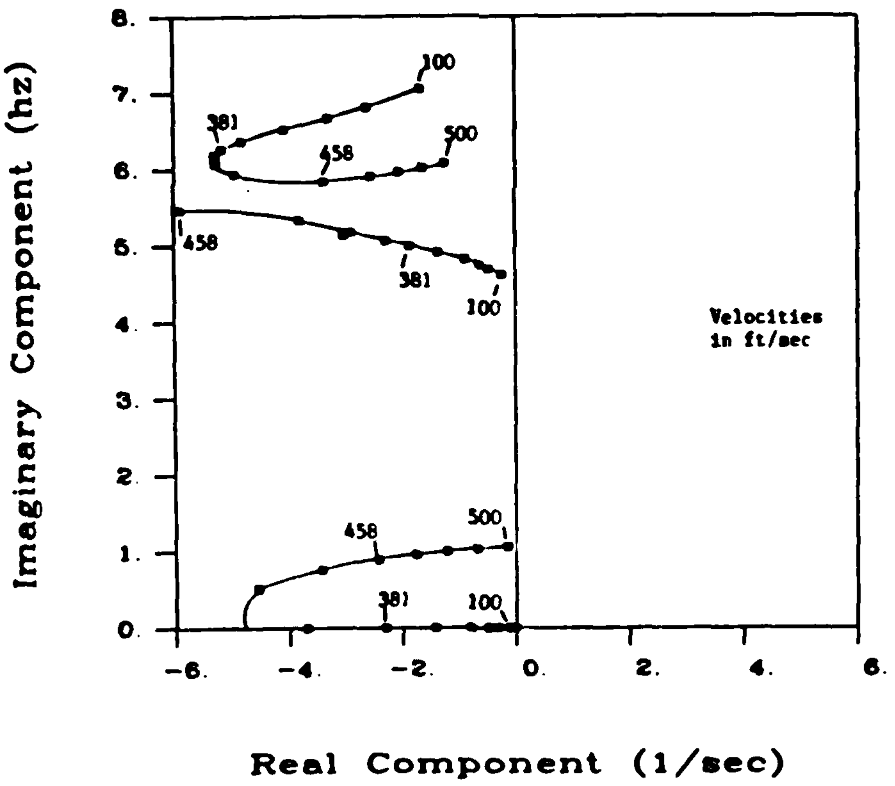


Figure 7. Velocity Root Locus for the YF-17 Model Using State-space Feedback

V. Conclusions and Recommendations

The equations of motion of an elastic aircraft, have been developed in this thesis, including the effects of control surface dynamics. As was shown, the model developed is useful for stability and control analysis, including the effects of structural dynamics. The model can be used as a basis for either classical or modern control methods. This model also has merit as a flutter prediction method, at least where the aerodynamics are well behaved, and can be linearly approximated at low reduced frequencies.

There are several advantages to using this method. First, stability derivatives can be predicted, but if a better source of stability derivatives is known, they can be directly incorporated into the model. A second advantage is that the states are the normal aircraft states and rates, plus those associated with the structural generalized coordinates and rates, and those associated with the control surface rotations and rates. This makes this method directly applicable to classical and modern control design procedures. A third advantage is that the second order approximation used in this thesis results in a lower order model than the methodology used in ADAM. Even if higher order dynamics are needed, they can be easily added to the model as pointed out by Rodden (20). Finally, this method can allow the prediction of control surface instabilities, and unlike ADAM can predict rigid body motion.

This project is a useful design tool and it should continue to be expanded. The use of other dynamic models should be used to further validate this method, including models in which the rigid modes couple with the structural modes. Second, the ability to extend this model to beyond second order, and to extend it to lateral-directional motion. Finally, the ability to account for sensor and actuator dynamics, along with control system dynamics needs to be addressed, to provide a better dynamic model.

A. YF-17 State-Space Model at 1.2 V_f

This appendix contains the A, B, and C matrices for the YF-17 model at 20 percent above the flutter speed (458 ft/sec). The states of the system are in Table A1. The A matrix is contained in Table A2. The B matrix is shown in Table A3, while the C matrix is in Table A4.

TABLE A1

THE STATES OF THE YF-17 STATE-SPACE MODEL

Z
θ
ξ_1-Structural mode
ξ_2-Structural mode
δ_1-Control surface mode
ξ_2-Control surface mode
dZ/dt
$d\theta/dt$
$d\xi_1/dt$
$d\xi_2/dt$
$d\delta_1/dt$
$d\delta_2/dt$

TABLE A2

THE A MATRIX OF THE YF-17 MODEL AT 458 FT/SEC

```
ROW  1
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.1000E+01 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
ROW  2
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.0000E+00 0.1000E+01 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
ROW  3
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.0000E+00 0.0000E+00 0.1000E+01 0.0000E+00 0.0000E+00 0.0000E+00
ROW  4
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.0000E+00 0.0000E+00 0.0000E+00 0.1000E+01 0.0000E+00 0.0000E+00
ROW  5
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.1000E+01 0.0000E+00
ROW  6
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.1000E+01
ROW  7
 0.0000E+00 0.0000E+00-0.3640E+02 0.8012E+02 0.1733E+03 0.6701E+03
-0.1972E+01 0.4510E+03-0.3167E+00-0.1976E-02 0.2958E+00 0.2355E+00
ROW  8
 0.0000E+00 0.0000E+00-0.2536E+00 0.3157E+01-0.5169E+02-0.2357E+03
-0.5530E-01-0.2834E+01-0.3462E-01 0.6292E-01-0.1353E-01-0.2464E+00
ROW  9
 0.0000E+00 0.0000E+00-0.1063E+04 0.5077E+03 0.8081E+03 0.5011E+04
-0.6238E+01 0.1225E+01-0.2530E+01 0.9873E+00 0.6515E+00 0.3356E+01
ROW 10
 0.0000E+00 0.0000E+00-0.2731E+03-0.1428E+04 0.2240E+04 0.6060E+04
-0.9393E+01 0.5419E+01-0.1407E+01-0.2632E+01 0.2659E+01 0.2462E+01
ROW 11
 0.0000E+00 0.0000E+00-0.6130E+01 0.1664E+02-0.9885E+05 0.2625E+02
-0.2628E+00-0.2952E+02-0.2426E+00 0.3485E+00-0.1015E+01 0.1170E+00
ROW 12
 0.0000E+00 0.0000E+00 0.4492E+02-0.4854E+02-0.1162E+03-0.1395E+06
 0.2328E+00-0.3660E+02 0.3252E+00 0.2543E+00-0.1188E+00-0.2577E+02
```

TABLE A3

THE B MATRIX OF THE YF-17 MODEL AT 458 FT/SEC

$$\begin{bmatrix}
0.0000E+00 & 0.0000E+00 \\
0.0000E+00 & 0.0000E+00 \\
0.0000E+00 & 0.0000E+00 \\
0.0000E+00 & 0.0000E+00 \\
0.0000E+00 & 0.0000E+00 \\
0.0000E+00 & 0.0000E+00 \\
-0.2679E+03 & -0.9205E+03 \\
0.1689E+03 & 0.3064E+03 \\
-0.1233E+04 & -0.4638E+04 \\
-0.2295E+04 & -0.6524E+04 \\
0.9885E+05 & -0.3273E+02 \\
0.1011E+03 & 0.1396E+06
\end{bmatrix}.$$

TABLE A4

THE C MATRIX OF THE YF-17 MODEL AT 458 FT/SEC

USING TWIST POSITION SENSOR

$$[0 \;\; 0 \;\; 0.008089 \;\; -0.01584 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0]$$

B. MAC User's Manual

The Methodology for Aeroelasticity and Controls (MAC) program was developed from the theory presented in the main body of this thesis. MAC integrates the results of unsteady aerodynamics codes, structural dynamics, and controls into a useful tool for both structural dynamics and flight control engineers. MAC was developed under AFIT thesis number AFIT/GAE/AA/86J-02 for the Flight Dynamics Laboratory. The program resides on the Flight Dynamics Laboratory Vax 11/785 computer with a VMS operating system. MAC accesses the IMSL library, the PLOT10 plotting library and the DI3000 plotting library. The example input data in this appendix is the YF-17 model data used in this thesis.

The majority of input to MAC can either be input from the terminal or automatically input from files. There are three exceptions to this. The aerodynamic forces and the print input data must be input from files, while the feedback matrices must be input from the terminal. The following logical file units are assigned to the following data files.

TABLE B1

DATA FILES USED IN PROGRAM MAC

Logical Unit	Data File Content
1	Structural Data
2	Aerodynamic Data
3	Initial Conditions
4	Output File
5	Terminal Input
6	Terminal Output
7	Root Locus Input
9	Output Files for Use in MATRIXx
10	Print Data
11	Forcing Function Matrix
12	C Matrix
90	Data for Plotting Root Locus
98	Scratch File
99	Scratch File

The first thing MAC does when started is to read the system print parameters from a file. Then the user will be prompted by the following menu.

```
MAC ready for input

 Enter (#):

     1.  Automatic input of items (2)-(7)
     2.  Read in structural data
     3.  Read in aerodynamic data
     4.  Read in initial conditions
     5.  Read in root locus values
     6.  Read in forcing function matrix
     7.  Read in the C matrix
     8.  Completion of input data
```

If the user chooses option 1, MAC will automatically read in all the data from items 2-7 from files. If the user wishes to input the data separately, the options 2-8 must be accomplished in order. Once the data is input, the remainder of the program will prompt the user for input. The terminal input is in free field format. The remaining portion of this manual describes the format of the input data on files.

Print Data. The system constants for the determination of what is to be output is in this file. The file is in namelist format, as shown in Table B2. Below is an explanation of each variable and it's default value.

NAERO(0)	- set to 1 to print the aerodynamic matrices onto unit 4, the output file
NMAT(0)	- set to 1 to print the X, Y, and Z matrices onto unit 4

NINV(0) - Set to 1 to print the X^{-1} matrix onto unit 4

NAMAT(0) - set to 1 if the A and B matrices are to be set to files that are compatible with MATRIXx

NEIG(1) - set to 0 if the eigenvalues are not to be sent to the plot file

NSTAB(0) - set to 1 if the rigid body non-dimensional stability derivatives are to be modified

TABLE B2

PRINT DATA EXAMPLE

```
$PRNT NAERO=0, NMAT=1, NINV=0, NAMAT=1, NEIG=1, NSTAB=0 $END
```

Structural Data. The structural data is input the diagonal elements of mass matrix, and the associated frequency (in Hz) of vibration. Table B3 contains an example. The data is input in the following format.

Line 1 - N=total # of modes (limited to 20), NRB=# of rigid modes (limited to 2), NS=# of structural modes (limited to 13), NC=# of control surface modes (limited to 5)
Format (4I5)

Line 2 - Generalized masses in the order of rigid modes, structural modes, and control surface modes. Line 2 is repeated as often as necessary to account for all the modes
Format (4E15.7)

Line 3 - Frequencies of the modes of vibration in the same order as above. Line 3 is also repeated as necessary
Format (4E15.7)

TABLE B3

EXAMPLE STRUCTURAL DATA

```
  6    2    2    2
0.7061300E+01  0.7776900E+02  0.3115000E+00  0.1019000E+00
0.1027000E+00  0.1000000E+00
0.0000000E+00  0.0000000E+00  0.4628000E+01  0.7186000E+01
0.5000000E+02  0.6000000E+00
```

<u>Aerodynamic</u> <u>Data</u>. The generalized aerodynamic forces (Q_{ij}s), from unsteady aerodynamic codes is input as complex pairs with j increasing before i. Table B4 is an example of this data for the YF-17. The format of the variables is shown below Table B4. This is all the data that MAC will use from this file. It will ignore any other aerodynamic data. K1 should be 0.0, and K2 should be a small value of reduced frequency.

<u>Initial</u> <u>Condition</u> <u>Data</u>. This data includes the density speed of sound, reference areas and lengths, and the mode shape magnitudes. Table B5 is an example of this data for the YF-17. The variable format is explained below.

Line 1 - RHOO=density used in slugs/ft^3, AO=speed of sound at that density altitude in ft/sec, SREF=reference wing area in ft^2, CBAR=reference chord (Mean Aerodynamic Chord) if ft, BR=reference semi-chord in ft
Format (F10.7, F10.2, 3F10.5)

Line 2 - ALPHAO and HO, the magnitudes of the pitch and

TABLE B4

EXAMPLE AERODYNAMIC DATA

```
YF-17 MODEL 6 MODES
    6          Ø.8ØØØØØØE+ØØ
  Ø.ØØØØØØØE+ØØ
  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ
  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.ØØØØØØØE+ØØ  Ø.68ØØ126E+Ø3  Ø.ØØØØØØØE+ØØ  Ø.294767ØE+Ø4  Ø.ØØØØØØØE+ØØ
  Ø.93926Ø9E+Ø2  Ø.ØØØØØØØE+ØØ  Ø.4584272E+Ø2  Ø.ØØØØØØØE+ØØ  Ø.1662773E+Ø2  Ø.ØØØØØØØE+ØØ -Ø.1Ø916Ø4E+Ø2  Ø.ØØØØØØØE+ØØ
  Ø.2258454E+Ø1  Ø.ØØØØØØØE+ØØ  Ø.3964384E+Ø1  Ø.ØØØØØØØE+ØØ  Ø.6Ø49513E+ØØ  Ø.ØØØØØØØE+ØØ  Ø.2395Ø11E+ØØ  Ø.ØØØØØØØE+ØØ
  Ø.7938793E-Ø1  Ø.ØØØØØØØE+ØØ -Ø.46Ø4Ø41E+ØØ  Ø.ØØØØØØØE+ØØ -Ø.5152788E+Ø1  Ø.ØØØØØØØE+ØØ -Ø.2444379E+Ø2  Ø.ØØØØØØØE+ØØ
 -Ø.143726ØE+Ø1  Ø.ØØØØØØØE+ØØ -Ø.5655986E+ØØ  Ø.ØØØØØØØE+ØØ -Ø.1919244E+ØØ  Ø.ØØØØØØØE+ØØ  Ø.52ØØ465E+ØØ  Ø.ØØØØØØØE+ØØ
  Ø.7166612E+Ø2  Ø.ØØØØØØØE+ØØ  Ø.1363Ø28E+Ø4  Ø.ØØØØØØØE+ØØ  Ø.14168Ø5E+Ø2  Ø.ØØØØØØØE+ØØ  Ø.55634Ø3E+ØØ  Ø.ØØØØØØØE+ØØ
  Ø.2Ø29873E+Ø2  Ø.ØØØØØØØE+ØØ  Ø.2244352E+Ø1  Ø.ØØØØØØØE+ØØ  Ø.189264ØE+Ø3  Ø.ØØØØØØØE+ØØ  Ø.1116695E+Ø5  Ø.ØØØØØØØE+ØØ
 -Ø.127119ØE+Ø2  Ø.ØØØØØØØE+ØØ  Ø.49ØØ815E+Ø1  Ø.ØØØØØØØE+ØØ  Ø.928923ØE+ØØ  Ø.ØØØØØØØE+ØØ -Ø.1121355E+Ø3  Ø.ØØØØØØØE+ØØ
  Ø.2ØØØØØØE-Ø1
 -Ø.8329Ø56E-Ø2  Ø.7629888E+ØØ -Ø.96Ø27Ø5E+ØØ  Ø.3337679E+Ø1  Ø.1852165E-Ø2  Ø.1Ø527Ø5E+ØØ  Ø.1172Ø3ØE-Ø2  Ø.5138736E-Ø1
 -Ø.1576Ø15E-Ø2  Ø.1866987E-Ø1 -Ø.6426724E-Ø2 -Ø.1235671E-Ø1  Ø.68ØØ139E+Ø3  Ø.3416318E+Ø2  Ø.2954977E+Ø4  Ø.173Ø164E+Ø4
  Ø.9387524E+Ø2  Ø.5243399E-Ø1  Ø.4582468E+Ø2 -Ø.2173455E+ØØ  Ø.1661179E+Ø2  Ø.2817868E+Ø1 -Ø.1171492E+Ø2  Ø.1113Ø99E+Ø2
  Ø.2258491E+Ø1  Ø.122Ø621E+ØØ  Ø.3969428E+Ø1  Ø.1857424E+Ø1  Ø.6Ø44776E+ØØ  Ø.431Ø363E-Ø1  Ø.2395887E+ØØ  Ø.7629235E-Ø2
  Ø.78999Ø2E-Ø1  Ø.1666135E-Ø1 -Ø.461ØØ35E+ØØ -Ø.21Ø5576E-Ø1 -Ø.51498Ø5E+Ø1 -Ø.15Ø7898E-Ø2 -Ø.2455194E+Ø2 -Ø.316218ØE+Ø1
 -Ø.1436477E+Ø1 -Ø.1753Ø61E-Ø1 -Ø.5651587E+ØØ  Ø.1446ØØ6E-Ø1 -Ø.19190Ø21E+ØØ -Ø.2368Ø65E-Ø1  Ø.519621ØE+ØØ -Ø.1687229E-Ø1
  Ø.71589Ø3E+Ø2 -Ø.135Ø853E+Ø1  Ø.1364436E+Ø4  Ø.693Ø7Ø6E+Ø1  Ø.1415249E+Ø2 -Ø.125977ØE+ØØ  Ø.5469Ø66E+ØØ -Ø.1737129E+ØØ
  Ø.2Ø29875E+Ø2  Ø.8237334E+ØØ  Ø.2248991E+Ø1  Ø.8775174E-Ø1  Ø.18Ø9755E+Ø3 -Ø.3ØØ1544E+ØØ  Ø.1117Ø21E+Ø5  Ø.1371389E+Ø3
 -Ø.1275363E+Ø2 -Ø.5139786E+ØØ  Ø.48818Ø6E+Ø1 -Ø.83Ø3293E-Ø1  Ø.9274798E+ØØ -Ø.9225371E-Ø1 -Ø.1121958E+Ø3  Ø.2Ø63767E+Ø2
```

Line 1 - TITLE=title of the aerodynamics, Format(80A)
Line 2   N=number of modes, XMACH=mach number of the aerodynamics, Format (I5,5X,E15.7)
Block 1
  Line 1 - K1=first reduced frequency (0), Format (E15.7)
  Line 2 - GF1(I,J)=first generalized aerodynamic matrix, input in complex pairs, with
           J coming before I. Line 2 is repeated as often as necessary.
           Format (4(2E15.7))
Block 2
  Line 1 - K2=second reduced frequency (small), Format (E15.7)
  LIne 2 - GF2(I,J)=second generalized aerodynamic matrix, input in complex pairs, with
           J coming before I. Line 2 is repeated as often as necessary.
           Format (4(2E15.7))

plunge mode shapes
Format (2F10.5)

Line 3 - ZETA(I)=the magnitudes of the structural mode shapes. This line is repeated as often as necessary to account for all the elastic modes
Format (6F10.5)

Line 4 - DELTA(I)=the magnitudes of the control surface mode shapes
Format (6F10.5)

TABLE B5

EXAMPLE INITIAL CONDITION DATA

```
 0.0010700   1100.00  13.68000   2.97000   1.48500
 12.00000   1.00000
  1.00000   1.00000
 12.00000  12.00000
```

Root Locus Data. The root locus data contains the velocities that are to be used to compute the eigenvalues for plotting. The maximum number is 20. Table B6 shows an example.

Line 1 - NVEL=.of velocities to be used
Format (I5)

Line 2 - V(I)=each velocity in ft/sec, repeated NVEL times
Format (F10.3)

TABLE B6

EXAMPLE ROOT LOCUS DATA

```
   15
     0.000
   100.000
   200.000
   250.000
   300.000
   350.000
   381.000
   400.000
   420.000
   440.000
   458.000
   470.000
   480.000
   490.000
   500.000
```

Forcing Matrix. The forcing matrix is used to create the B matrix. It is defined in the main text of the thesis. Table B7 is the example used for the YF-17.

Line 1 -	NINP=#of inputs the system Format (I5)
Line 2 -	QMAT(I,J)=the forcing matrix (NxNINP) input row first, j increases before i. Line 2 is repeated as often as necessary. Format (6E15.7)

TABLE B7

EXAMPLE FORCING MATRIX

```
  1
0.0000000E+00  0.0000000E+00  0.8089000E-02 -0.1584000E-01  0.0000000E+00  0.0000000E+00
0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
```

Measurement Matrix. The C matrix is input from this file. Table B8 is the example used for the YF-17.

Line 1 - NOUT=#of outputs into the system
Format (I5)

Line 2 - CMAT(I,J)=the measurement matrix (NOUTxN) input row first, j increases before i. Line 2 is repeated as often as necessary.
Format (6E15.7)

TABLE B8

EXAMPLE MEASUREMENT MATRIX

```
  2
0.0000000E+00  0.0000000E+00  0.1000000E+01  0.1000000E+01  0.0000000E+00  0.0000000E+00
0.0000000E+00  0.0000000E+00  0.1000000E+01  0.0000000E+00  0.0000000E+00  0.1000000E+01
```

The remaining inputs are from the terminal, as requested by the the program prompts.

C. MAC Program

This Appendix contains the program listings of all the programs that comprise the MAC program. MAC uses the IMSL subroutine library for matrix inversion (LINV1F), matrix multiplication (VMULFF), and eigenvalue computation (EIGRF). MAC also uses the PLOT10 plot routine library in the velocity root locus plotting program to clear the screen before plotting. Finally, MAC uses the DI3000 plot routine library to create both screen plots and the hard copy plots. The main calling program (MAC) is listed first. The remaining subroutine libraries are in alphabetical order. All the routines are listed below as they appear in this appendix.

TABLE C1

MAC PROGRAM LISTING ORDER

MAC	INPUT
AMAT	MATSAV
AUGMENT	MINV
BMAT	MMULT
EVAL	RLPLOT
FASTCHG	STABDER

The subroutines are fairly well commented so that a person familiar with the theory developed in the main text and with FORTRAN should be able to interpret the program.

```
      PROGRAM MAC
C
C  This program was developed under AFIT Thesis AFIT/GAE/AA/86J-2
C by John J. Cerra II.
C
C  This program uses generalized mass and stiffness matrices,
C along with generalized forces from an external source (FASTOP)
C and combines them into an appropriate state space formulation for
C use by any control analysis and design package.  The state space
C model is the minimum order attainable, 2N, where N is the number
C of modes used (including rigid body and control surface modes).
C
C  Subroutine INPUT reads in the generalized mass and stiffness
C matrices.  It also reads in the generalized forces from FASTOP at
C two reduced frequencies (usually k=0 and a small k).  The initial
C conditions are also input here, along with the root locus data.
C Finally it also inputs the forcing function matrix, Q,
C and the C matrix if needed.
C
C  Subroutine STABDER computes stability derivatives from the FASTOP
C unsteady aerodynamics.  There is also an option to modify the
C derivatives if derivatives from a better source are known.
C
C  Subroutine AMAT uses the above data to compute the state space
C A matrix used for stability analysis.  The order of A is 2N.  A
C is left as a function of dynamic pressure so that a velocity
C root locus can be performed as in a typical flutter solution.
C
C  Subroutine BMAT computes the B matrix from the forcing function
C matrix Q.  B is a function of dynamic pressure.
C
C  Subroutine MINV, called by AMAT, computes the inverse of the X
C matrix using an IMSL routine LINV1F.
C
C  Subroutine MMULT, called by AMAT, and BMAT multiplies matrices.
C This is used to form the A and B matrices.
C
C  Subroutine EVAL computes the eigenvalues of the A matrix
C The eigenvalues for various velocities can
C be accomplished and then can be plotted on the real and imaginary
C plane to produce a velocity root locus.
C
C  Subroutine AUGMENT augments the A matrix with the appropriate
C feedback, to form the augmented (closed loop) A matrix
C
C  Subroutine FASTCHG creates non-dimensional coefficients out of
C the unsteady aerodynamics of FASTOP
C
C  Subroutine RLPLOT plots the eigenvalues of the A matrix with respect
C to velocity on the complex plane, creating a 'velocity root locus'.
```

```
C
C  Subroutine MATSAV saves matrices to a file in a compatible
C form for MATRIXx
C
      CHARACTER*1 QAUG,QLPT
      CHARACTER*80 TITLE
      INTEGER NVEL
      REAL VE,V(20)
      COMMON/RL/NVEL,V
C
C  Open output file
C
      OPEN(4,STATUS='NEW')
      CALL INPUT
      OPEN(98,STATUS='SCRATCH')
      WRITE(98,*) NVEL
C
C Velocity root locus loop for the unaugmented aircraft
C
      IF (NVEL.GT.1) THEN
       DO 1 I=1,NVEL
        CALL STABDER(NVEL,V(I))
        CALL AMAT
        CALL BMAT(V(I))
        CALL EVAL
    1  CONTINUE
      ELSE
       NVEL=0
       VE=0.001
       CALL STABDER(NVEL,VE)
       CALL AMAT
       CALL BMAT(VE)
       CALL EVAL
      ENDIF
C
C  Option to plot the open loop  aircraft roots
C
      WRITE(*,'(A)') ' Do you want to plot the open loop roots?'
      READ(*,100) QPLT
      IF (QPLT.EQ.'Y') THEN
       WRITE(*,'(A)') ' What title do you want for the plot?'
       READ(*,100) TITLE
       CALL RLPLOT(0.0,8.0,-6.0,6.0,TITLE)
       CLOSE(90)
      ENDIF
      WRITE(*,'(A)') ' Do you want to augment the A matrix?'
      READ(*,100) QAUG
      IF (QAUG.EQ.'Y') THEN
C
C  Call to closed lop augmentation program
C
```

```
       CALL AUGMENT
      ENDIF
      IDAT=99
      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
      CLOSE(4)
      CLOSE(7)
      CLOSE(90)
      CLOSE(9)
      CLOSE(10)
      CLOSE(11)
      CLOSE(12)
      CLOSE(98)
      WRITE(*,'(A)') ' This program has ended.'
  100 FORMAT(A)
      END
      SUBROUTINE AMAT
C
C  This subroutine uses the stability derivatives above for use in
C developing the open loop reduced order A matrix for stability
C analysis, and control work.  It is of much smaller order than
C that resulting from using Pade fit aerodynamics.
C
      CHARACTER*1 QASAV
      CHARACTER*20 ANAM
      INTEGER N,NRB,NS,NC,NV
      COMMON/INDEX/N,NRB,NS,NC
      INTEGER NAERO,NMAT,NINV,NAMAT,NEIG
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG
      REAL ZALPHA,MALPHA,ZALPHDT,MALPHDT,ZQ,MQ,
     @FZALPHA,FZALPDTN,FZETQ,ZZETA,MZETA,
     @ZZETADT,MZETADT,ZZETDDT,MZETDDT,
     @FZETZET,FZETZDT,FZEZDDT,
     @FZETDEL,FZETDDT,FZEDDDT,
     @FDELZET,FDELZDT,FDEZDDT,
     @ZDELTA,MDELTA,FDALPHA,FDALPDT,FDETQ,
     @ZDELTDT,MDELTDT,ZDELDDT,MDELDDT,
     @FDELDEL,FDELDDT,FDEDDDT,VEL
      DIMENSION FZALPHA(15),FZALPDT(15),FZETQ(15),ZZETA(15),MZETA(15),
     @ZZETADT(15),MZETADT(15),ZZETDDT(15),MZETDDT(15),
     @FZETZET(15,15),FZETZDT(15,15),FZEZDDT(15,15),
     @FZETDEL(15,5),FZETDDT(15,5),FZEDDDT(15,5),
     @FDELZET(5,15),FDELZDT(5,15),FDEZDDT(5,15),
     @ZDELTA(5),MDELTA(5),FDALPHA(5),FDALPDT(5),FDETQ(5),
     @ZDELTDT(5),MDELTDT(5),ZDELDDT(5),MDELDDT(5),
     @FDELDEL(5,5),FDELDDT(5,5),FDEDDDT(5,5)
      COMMON/DERIV/ZALPHA,MALPHA,ZALPHDT,MALPHDT,ZQ,MQ,FZALPHA,FZALPDT,
     @FZETQ,ZZETA,MZETA,ZZETADT,MZETADT,ZZETDDT,MZETDDT,FZETZET,
     @FZETZDT,FZEZDDT,FZETDEL,FZETDDT,FZEDDDT,FDELZET,FDELZDT,FDEZDDT,
     @ZDELTA,MDELTA,FDALPHA,FDALPDT,FDETQ,
     @ZDELTDT,MDELTDT,ZDELDDT,MDELDDT,FDELDEL,FDELDDT,FDEDDDT,VEL
      REAL GM,GK
      DIMENSION GM(20),GK(20)
      COMMON/STRUCT/GM,GK
      REAL K1,K2
      COMPLEX GF1,GF2
      DIMENSION GF1(20,20),GF2(20,20
      COMMON/AERO/K1,K2,GF1,GF2,PI
      REAL X,Y,Z,AMATRIX,XINV,XINVY,XINVZ
      DIMENSION X(20,20),Y(20,20),Z(20,20),AMATRIX(40,40),XINV(40),
     @XINVY(20,20),XINVZ(20,20)
      COMMON/INV/X,XINV
      COMMON/MMUL/Y,Z,XINVY,XINVZ
      COMMON/EIG/AMATRIX
      REAL XMOM(5)
      COMMON/MOMENT/XMOM
      COMPLEX E(20)
      DIMENSION WKX(20)
      REAL*8 A(40,40),DUMMY
      N2=N*2
C
C  Initialization of the matrices of: Xx+Yx+Zx=0.
C
      DO 30 I=1,N
       DO 30 J=1,N
        X(I,J)=0.0
        Y(I,J)=0.0
        Z(I,J)=0.0
   30 CONTINUE
C
C Input of the structural information into the matrices.
C
      DO 1 I=1,N
       X(I,I)=GM(I)
       Z(I,I)=GK(I)
    1 CONTINUE
C
C  Input of the aerodynamics into the matrices.
C
      IF(NRB.EQ.0) GOTO 24
C
C  Rigid aero
C
      X(1,1)=X(1,1)-ZALPHDT/VEL
      X(2,1)=-MALPHDT/VEL
      Y(1,1)=-ZALPHA/VEL
      Y(2,1)=-MALPHA/VEL
      Y(1,2)=-ZQ-GM(1)*VEL
      Y(2,2)=-MQ
   24 CONTINUE
      IF (NS.EQ.0) GOTO 21
C
C Elastic/Rigid and Rigid/Elastic aero
C
      DO 2 I=1,NS
       J=I+NRB
       IF (NRB.EQ.0) GOTO 20
       X(J,1)=-FZALPDT(I)/VEL
       X(1,J)=-ZZETDDT(I)
       X(2,J)=-MZETDDT(I)
       Y(J,1)=-FZALPHA(I)/VEL
       Y(J,2)=-FZETQ(I)
       Y(1,J)=-ZZETADT(I)
       Y(2,J)=-MZETADT(I)
       Z(1,J)=-ZZETA(I)
       Z(2,J)=-MZETA(I)
   20 CONTINUE
```

```
        IF (NS.EQ.0) GOTO 4
C
C  Elastic/Elastic aero
C
        DO 4 K=1,NS
         L=K+NRB
         X(J,L)=-FZEZDDT(I,K)+X(J,L)
         Y(J,L)=-FZETZDT(I,K)
         Z(J,L)=-FZETZET(I,K)+Z(J,L)
    4   CONTINUE
       IF (NC.EQ.0) GOTO 21
C
C  Elastic/Control and Control/Elastic aero
C
       DO 5 K=1,NC
         L=K+NS+2
         X(L,J)=-FDEZDDT(K,I)
         X(J,L)=-FZEDDDT(I,K)
         Y(L,J)=-FDELZDT(K,I)
         Y(J,L)=-FZETDDT(I,K)
         Z(L,J)=-FDELZET(K,I)
         Z(J,L)=-FZETDEL(I,K)
    5   CONTINUE
   21 CONTINUE
    2 CONTINUE
       IF (NRB.EQ.0) GOTO 23
       IF (NC.EQ.0) GOTO 22
C
C Control/Rigid, Rigid/Control aero
C
       DO 3 I=1,NC
        J=I+NRB+NS
        X(J,1)=-FDALPDT(I)/VEL
        X(1,J)=-ZDELDDT(I)
        X(2,J)=-MDELDDT(I)
        Y(J,1)=-FDALPHA(I)/VEL
        Y(J,2)=-FDETQ(I)
        Y(1,J)=-ZDELTDT(I)
        Y(2,J)=-MDELTDT(I)
        Z(1,J)=-ZDELTA(I)
        Z(2,J)=-MDELTA(I)
   23 CONTINUE
        IF (NC.EQ.0) GOTO 6
C
C  Control/Control aero
C
        DO 6 K=1,NC
         L=K+NS+NRB
         X(J,L)=-FDEDDDT(I,K)+X(J,L)
         Y(J,L)=-FDELDDT(I,K)
         Z(J,L)=-FDELDEL(I,K)+Z(J,L)
    6  CONTINUE
    3 CONTINUE
      DO 32 J=1,NC
       L=J+NS+NRB
       XMOM(J)=Z(L,L)
   32 CONTINUE
   22 CONTINUE
      IF (NMAT.EQ.0) GOTO 14
      WRITE(4,'(/A)') ' The X matrix is'
      DO 10 I=1,N
       WRITE(4,300) I
       DO 11 J=1,N,6
        WRITE(4,301) (X(I,K),K=J,J+5)
   11  CONTINUE
   10 CONTINUE
      WRITE(4,'(/A)') ' The Y matrix is'
      DO 12 I=1,N
       WRITE(4,300) I
       DO 13 J=1,N,6
        WRITE(4,301) (Y(I,K),K=J,J+5)
   13  CONTINUE
   12 CONTINUE
      WRITE(4,'(/A)') ' The Z matrix is'
      DO 14 I=1,N
       WRITE(4,300) I
       DO 15 J=1,N,6
        WRITE(4,301) (Z(I,K),K=J,J+5)
   15  CONTINUE
   14 CONTINUE
C
C  Initialization of the A matrix.
C
      DO 31 I=1,N2
       DO 31 J=1,N2
        AMATRIX(I,J)=0.0
   31 CONTINUE
C
C  Call to invert X and to multiply XINV*Y and XINV*Z
C
      CALL MINV
      CALL MMULT
      DO 8 I=1,N
      DO 8 J=1,N
       K=I+N
       L=J+N
       AMATRIX(K,L)=-XINVY(I,J)
       AMATRIX(I,K)=1.0
       AMATRIX(K,J)=-XINVZ(I,J)
    8 CONTINUE
      IF (NAMAT.EQ.1) THEN
      OPEN(99,STATUS='SCRATCH')
      NV=NINT(VEL)
      WRITE(99,310) NV
  310 FORMAT('[.MATRXX]A',I4.4)
      REWIND 99
      READ(99,302) ANAM
      CLOSE(99)
      DO 40 I=1,N2
       DO 40 J=1,N2
        AIJ=AMATRIX(I,J)
        A(I,J)=DBLE(AIJ)
   40 CONTINUE
      OPEN(9,FILE=ANAM,STATUS='NEW')
C
C  Call to the routine to put the A matrix in MATRIXx format
C
      CALL MATSAV(9,ANAM,40,N2,N2,0,A,DUMMY,'(1P8E15.7)')
      CLOSE(9)
      ENDIF
      WRITE(4,'(/A)') ' The A matrix of dx/dt=A*x is'
      DO 16 I=1,N2
       WRITE(4,300) I
       DO 17 J=1,N2,6
        WRITE(4,301) (AMATRIX(I,K),K=J,J+5)
   17  CONTINUE
   16 CONTINUE
      WRITE(98,*) ((AMATRIX(I,J),J=1,N2),I=1,N2)
      IF (NRB.NE.0) WRITE(4,304)
      IF (NS.NE.0) WRITE(4,305) (I,I=1,NS)
      IF (NC.NE.0) WRITE(4,306) (I,I=1,NC)
      IF (NRB.NE.0) WRITE(4,307)
      IF (NS.NE.0) WRITE(4,308) (I,I=1,NS)
      IF (NC.NE.0) WRITE(4,309) (I,I=1,NC)
  300 FORMAT(' ROW',I3)
  301 FORMAT(1X,6E11.4)
  302 FORMAT(A)
  304 FORMAT(/,' The states of the A matrix are:',/,
     0' Z',/,
     0' THETA')
  305 FORMAT(' ZETA(',I2,')-Structural mode')
  306 FORMAT(' DELTA(',I2,')-Control surface mode')
  307 FORMAT(' dZ/dt',/,' dTHETA/dt')
  308 FORMAT(' dZETA(',I2,')/dt')
  309 FORMAT(' dDELTA(',I2,')/dt')
  303 FORMAT(6E15.7)
      RETURN
      END
      SUBROUTINE AUGMENT
C
C   This subroutine will augment the state space equations with
C the appropriate feedback (output or state) to form the closed
C loop state equations
C
      REAL AUG(40,40),ACL(40,40),TEMP(40,40),CMAT(40,40),
     0BMATRIX(40,40),FMAT(5,40),V(20)
      INTEGER N,N2,NRB,NS,NC,IFDBK,NP,NOUT,IER,NA,NB
      CHARACTER*1 QPLT,QCHG
      CHARACTER*80 TITLE
      COMMON/EIG/ACL
      COMMON/INDEX/N,NRB,NS,NC
      COMMON/CMATRX/NOUT,CMAT
      COMMON/BMATX/NP,BMATRIX
      COMMON/RL/NVEL,V
      N2=2*N
    3 CONTINUE
      DO 8 I=1,5
       DO 8 J=1,40
        FMAT(I,J)=0.0
    8 CONTINUE
C
C  Find out whether output or state feedback is to be used
C
      WRITE(*,700)
      READ(*,701) IFDBK
      IF (IFDBK.GT.2) GOTO 3
      IF (IFDBK.EQ.0) GOTO 7
      IF (IFDBK.EQ.1) THEN
C
C  output feedback augmentation loop, creates BKC=AUG
C
       WRITE(*,702) NP,NOUT
       NA=NP
       NB=NOUT
       DO 1 I=1,NP
        WRITE(*,703) I
        READ(*,*) (FMAT(I,J),J=1,NOUT)
    1  CONTINUE
       GOTO 13
   14  CONTINUE
C
C Call to IMSL routine to multiply B*K, and then BK*C
C
       CALL VMULFF(BMATRIX,FMAT,N2,NP,NOUT,40,5,TEMP,40,IER)
       CALL VMULFF(TEMP,CMAT,N2,NOUT,N2,40,40,AUG,40,IER)
       WRITE(4,'(A)') ' For OUTPUT feedback'
       WRITE(4,'(A)') ' With a feedback matrix of'
       DO 16 I=1,NP
```

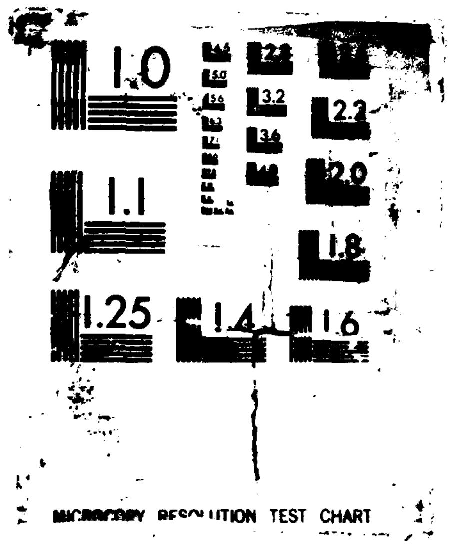
```
         WRITE(4,703) I
         DO 17 J=1,NOUT,6
          WRITE(4,706) (FMAT(I,K),K=J,J+5)
    17   CONTINUE
    16  CONTINUE
       ELSE
C
C  state feedback augmentation loop, creates BK=AUG
C
        WRITE(*,702) NP,N2
        NA=NP
        NB=N2
        DO 2 I=1,NP
         WRITE(*,703) I
         READ(*,*) (FMAT(I,J),J=1,N2)
     2  CONTINUE
        GOTO 13
    15  CONTINUE
C
C Call to IMSL routine to multiply B*K
C
        CALL VMULFF(BMATRIX,FMAT,N2,NP,N2,40,5,AUG,40,IER)
        WRITE(4,'(A)') ' For STATE feedback'
        WRITE(4,'(A)') ' With a feedback matrix of'
        DO 18 I=1,NP
         WRITE(4,703) I
         DO 19 J=1,N2,6
          WRITE(4,706) (FMAT(I,K),K=J,J+5)
    19   CONTINUE
    18  CONTINUE
       ENDIF
       DO 4 J=1,N2
        DO 4 I=1,N2
         TEMP(I,J)=0.0
     4 CONTINUE
C
C Unit 98 contains the unaugmented A matrices created during
C  the first root locus.  This section now creates another
C  root locus loop, computing eigenvalues and using the
C  closed loop A matrix Acl=A-AUG
C
       REWIND (98)
       READ(98,*) NVEL
       DO 5 K=1,NVEL
        READ(98,*) ((TEMP(I,J),J=1,N2),I=1,N2)
        DO 6 I=1,N2
         DO 6 J=1,N2
          ACL(I,J)=TEMP(I,J)-AUG(I,J)
     6  CONTINUE
        WRITE(*,704) V(K)
        WRITE(4,704) V(K)
       CALL EVAL
     5 CONTINUE
C
C Option to plot the root locus points
C
      WRITE(*,'(A)') ' Do you want to plot the closed loop roots?'
      READ(*,705) QPLT
      IF(QPLT.EQ.'Y') THEN
       WRITE(*,'(A)') ' What title do you want for the plot?'
       READ(*,705) TITLE
       CALL RLPLOT (0.0,8.0,-6.0,6.0,TITLE)
       CLOSE(90)
      ENDIF
      GOTO 3
C
C This section gives the user the option to change the values
C  of any portion of the feedback matrix.
C
   13 CONTINUE
      DO 10 I=1,NA
       WRITE(*,703) I
       DO 11 J=1,NB,6
        WRITE(*,706) (FMAT(I,K),K=J,J+5)
   11  CONTINUE
   10 CONTINUE
      WRITE(*,'(A)') ' Do you want to change any values?'
      READ(*,705) QCHG
      IF (QCHG.EQ.'N') GOTO 12
    9 CONTINUE
      WRITE(*,'(A)') ' Please input the row, column, and new value.'
      READ(*,*) I,J,FMAT(I,J)
      WRITE(*,'(A)') ' Do you want to change another value?'
      READ(*,705) QCHG
      IF (QCGG.EQ.'Y') GOTO 9
   12 CONTINUE
      IF (IFDBK.EQ.1) GOTO 14
      IF (IFDBK.EQ.2) GOTO 15
  700 FORMAT(' What type of feedback are you using?',//,
     @'     0.   Return to main program',/,
     @'     1.   Output',/,
     @'     2.   State',/)
  701 FORMAT(I5)
  702 FORMAT(' The feedback matrix will have',I3,' rows by',
     @I3,' columns.',/,' Please input the feedback matrix by rows.',/)
  703 FORMAT(' Row',I3)
  704 FORMAT(//,' The closed loop eigenvalues for ',F10.3,' ft/sec')
  705 FORMAT(A)
  706 FORMAT(1X,6E11.4)
    7 CONTINUE
      RETURN
      END
      SUBROUTINE BMAT(VEL)
C
C  This subroutine takes the forcing function matrix and converts it
C into the B matrix of dx/dt=Ax+Bu where u is the control input vector
C
      REAL X(20,20),XINV(20,20),QMAT(20,5),BMATRIX(40,5),BMAT1(20,5)
      INTEGER N,NRB,NS,NC,NINP,NAERO,NMAT,NINV,NAMAT,NEIG,N2,NV
      REAL*8 B(40,40),DUMMY
      CHARACTER*20 BNAM
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG,NV
      COMMON/INDEX/N,NRB,NS,NC
      COMMON/BMATRX/NINP,QMAT
      COMMON/INV/X,XINV
      COMMON/BMATX/NP,BMATRIX
      REAL XMOM(5)
      COMMON/MOMENT/XMOM
C
C  Call to IMSL routine to multiply XINV*QMAT
C
      CALL VMULFF(XINV,QMAT,N,N,NINP,20,20,BMAT1,20,IER)
      NP=NINP
C
C  Creates the actual B matrix
C
      DO 2 I=1,N
       K=N+I
       DO 2 J=1,NINP
        BMATRIX(K,J)=BMAT1(I,J)*XMOM(J)
    2 CONTINUE
      WRITE(4,603)
      N2=2*N
      DO 1 I=1,N2
       WRITE(4,600) I
       WRITE(4,601) (BMATRIX(I,J),J=1,NINP)
    1 CONTINUE
      IF (NAMAT.EQ.1) THEN
      OPEN(99,STATUS='SCRATCH')
      NV=NINT(VEL)
      WRITE(99,604) NV
  604 FORMAT('[.MATRXX]B',I4.4)
      REWIND 99
      READ(99,605) BNAM
  605 FORMAT(A)
      CLOSE(99)
       DO 10 I=1,N2
        DO 10 J=1,NINP
         BIJ=BMATRIX(I,J)
         B(I,J)=DBLE(BIJ)
   10  CONTINUE
```

```
        OPEN(9,FILE=BNAM,STATUS='NEW')
C
C  Call to routine that puts the B matrix into MATRIXx format
C
        CALL MATSAV(9,BNAM,40,N2,NINP,0,B,DUMMY,'(1P8E15.7)')
        CLOSE(9)
       ENDIF
  600 FORMAT(' Row',I3)
  601 FORMAT(1X,6E12.4)
  602 FORMAT(6E15.7)
  603 FORMAT(//,' The B matrix of dx/dt=Ax+Bu is')
      RETURN
      END
      SUBROUTINE EVAL
C
C  This subroutine computes the eigenvalues of the A matrix
C using the IMSL routine EIGRF.
C
      REAL AMATRIX,WK
      DIMENSION AMATRIX(40,40),WK(40)
      COMMON/EIG/AMATRIX
      COMPLEX EIGVAL(40)
      INTEGER N,NRB,NS,NC,IER,N2
      COMMON/INDEX/N,NRB,NS,NC
      INTEGER NAERO,NMAT,NINV,NAMAT,NEIG
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG
      CHARACTER*1 QEVAL
      CHARACTER*64 EIGFILE
      REAL*8 XX,XY,XZ
      N2=N*2
      PI=3.14159
C
C  Call to IMSL eigenvalue solver routine
C
      CALL EIGRF(AMATRIX,N2,40,0,EIGVAL,Z,IZ,WK,IER)
      IF (IER.NE.0) THEN
       WRITE(*,600) IER
       STOP
      ENDIF
      IF (NEIG.EQ.1) OPEN(90,STATUS='SCRATCH',FORM='UNFORMATTED')
      WRITE(*,'(/A)') ' The eigenvalues are:'
      WRITE(*,'(A)') '  I Eigenvalue             Freq(Hz)'
      WRITE(4,'(/A)') ' The eigenvalues are:'
      WRITE(4,'(A)') '  I Eigenvalue            Freq(Hz)'
       DO 2 I=1,N2
       FREQ=AIMAG(EIGVAL(I))/(2.*PI)
       XY=DBLE(FREQ)
       RV=REAL(EIGVAL(I))
       XX=DBLE(RV)
       XIM=AIMAG(EIGVAL(I))
       XZ=DBLE(XIM)
       WRITE(*,602) I,REAL(EIGVAL(I)),AIMAG(EIGVAL(I)),FREQ
       WRITE(4,602) I,REAL(EIGVAL(I)),AIMAG(EIGVAL(I)),FREQ
       IF (NEIG.EQ.1) WRITE(90) XX,XY,XZ
    2 CONTINUE
    1 CONTINUE
      WRITE(4,'(A)') '1'
  600 FORMAT(///,'**** There was an error in the eigenvalue',
     0' computation, # ',I3,' ****',///)
  601 FORMAT(A)
  602 FORMAT(I3,2E11.4,'i',E11.4)
      RETURN
      END
      SUBROUTINE FASTCHG
C
C This subroutine changers FASTOP generalized forces into
C  non-dimensional form by multiplying by s**2 and by dividing
C  by SREF for forces and by SREF*CBAR for moments
C  also the difference in the coordinate systems is taken into
C  account.  FASTOP is a left-hand rule axis system with positive
C  Z up and positive moment down.  Normal stability in body axes
C  positive Z is downward and positive moment is up.
C
      COMPLEX GF1,GF2
      REAL S,SREF,CBAR,PI,SR,CB,K1,K2,VO,RHOO,ALPHAO,HO,ZETAO,
     0DELTAO,BR
      INTEGER N,NRB,NS,NC
      DIMENSION GF1(20,20),GF2(20,20),ZETAO(15),DELTAO(5)
      COMMON/AERO/K1,K2,GF1,GF2,PI
      COMMON/IC/VO,RHOO,SREF,CBAR,ALPHAO,HO,ZETAO,DELTAO,BR
      COMMON/INDEX/N,NRB,NS,NC
      NRB1=NRB+1
      NSNRB=NS+NRB
      NSNRB1=NSNRB+1
      NCNSNRB=NSNRB+NC
      S=12.0
      SR=SREF*144.0
      CB=CBAR*12.0
      IF (NRB.EQ.0) GOTO 1
      DO 1 J=1,N
       GF1(1,J)=-GF1(1,J)*S**2/SR
       GF2(1,J)=-GF2(1,J)*S**2/SR
       GF1(2,J)=-GF1(2,J)*S**3/(SR*CB)
       GF2(2,J)=-GF2(2,J)*S**3/(SR*CB)
    1 CONTINUE
      IF (NS.EQ.0) GOTO 2
      DO 2 I=NRB1,NSNRB
       DO 2 J=1,N
        GF1(I,J)=-GF1(I,J)*S**2/SR
        GF2(I,J)=-GF2(I,J)*S**2/SR
    2 CONTINUE
      IF (NC.EQ.0) GOTO 3
      DO 3 I=NSNRB1,NCNSNRB
       DO 3 J=1,N
        GF1(I,J)=-GF1(I,J)*S**3/(SR*CB)
        GF2(I,J)=-GF2(I,J)*S**3/(SR*CB)
    3 CONTINUE
      RETURN
      END
      SUBROUTINE INPUT
C
C  This subroutine contains all the input necsssary to run the program.
C The total dimensions are 20 total modes, 2 rigid body modes, 5 control
C surface modes and 15 elastic modes maximum.
C
      INTEGER N,NRB,NS,NC,NSQ,NVEL,INPT,NINP,N2,NOUT
      INTEGER NAERO,NMAT,NINV,NAMAT,NEIG,NSTAB
      CHARACTER*64 STRUCT,AERO,INCOND,RTLCSF,INPTM,TITLE,OUTPM
      CHARACTER*1 Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11
      REAL K1,K2,VO,RHOO,SREF,CBAR,ALPHAO,HO,XMACH,AO,BR
      REAL GM,GK,ZETAO,DELTAO,OMEGA,V,F(20,20),G(20,20)
      REAL QMAT(20,5),CMAT(40,40)
      DIMENSION GM(20),GK(20),ZETAO(15),DELTAO(5),OMEGA(20),V(20)
      COMPLEX GF1,GF2
      DIMENSION GF1(20,20),GF2(20,20)
      COMMON/STRUCT/GM,GK
      COMMON/AERO/K1,K2,GF1,GF2,PI
      COMMON/INDEX/N,NRB,NS,NC
      COMMON/IC/VO,RHOO,SREF,CBAR,ALPHAO,HO,ZETAO,DELTAO,BR
      COMMON/RL/NVEL,V
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG
      COMMON/BMATRX/NINP,QMAT
      COMMON/CMATRX/NOUT,CMAT
      NAMELIST/PRNT/NAERO,NMAT,NINV,NAMAT,NEIG,NSTAB
C
C  Default values for the namelist variables
C
      NAERO=0
      NMAT=0
      NINV=0
      NAMAT=0
      NEIG=1
      NSTAB=0
      OPEN(10,STATUS='OLD')
      READ(10,PRNT)
      PI=3.14159
C
C  Question to find out whether the data is to be automatically
C input, or whether it will be done manually.
C
   24 CONTINUE
      WRITE(*,120)
      READ(*,121) INPT
      IF (INPT.EQ.1) THEN
C
C  Auto input of Structural info
C
       OPEN(1,STATUS='OLD')
       READ(1,116) N,NRB,NS,NC
      N2=2*N
      READ(1,117) (GM(I),I=1,N)
      READ(1,117) (OMEGA(I),I=1,N)
C
C  Auto input of Aero
C
      OPEN(2,STATUS='OLD')
      READ(2,111) TITLE
      READ(2,128) N,XMACH
      READ(2,127) K1
      READ(2,126) (((F(I,J),G(I,J)),I=1,N),J=1,N)
      DO 7 I=1,N
       DO 7 J=1,N
        GF1(I,J)=CMPLX(F(I,J),G(I,J))
    7 CONTINUE
      READ(2,127) K2
      READ(2,126) (((F(I,J),G(I,J)),I=1,N),J=1,N)
      DO 11 I=1,N
       DO 11 J=1,N
        GF2(I,J)=CMPLX(F(I,J),G(I,J))
   11 CONTINUE
C
C  Auto input of initial conditions
C
      OPEN(3,STATUS='OLD')
      READ(3,118) RHOO,AO,SREF,CBAR,BR
      IF (NRB.NE.0) READ(3,119) ALPHAO,HO
      IF (NS.NE.0) READ(3,119) (ZETAO(I),I=1,NS)
      IF (NC.NE.0) READ(3,119) (DELTAO(I),I=1,NC)
C
C  Auto input of root locus info
C
      OPEN(7,STATUS='OLD')
      READ(7,121) NVEL
      DO 15 I=1,NVEL
      READ(7,122) V(I)
   15 CONTINUE
C
C  Auto input of forcing function matrix
C
      OPEN(11,STATUS='OLD')
      READ(11,121) NINP
      READ(11,123) ((QMAT(I,J),J=1,NINP),I=1,N)
C
C  Auto input of C matrix
C
      OPEN(12,STATUS='OLD')
      READ(12,121) NOUT
      READ(12,123) ((CMAT(I,J),J=1,N2),I=1,NOUT)
      GOTO 23
     ENDIF
```

```
      IF (INPT.LE.0) GOTO 24
      IF (INPT.EQ.2) GOTO 20
      IF (INPT.EQ.3) GOTO 21
      IF (INPT.EQ.4) GOTO 22
      IF (INPT.EQ.5) GOTO 26
      IF (INPT.EQ.6) GOTO 25
      IF (INPT.EQ.7) GOTO 27
      IF (INPT.EQ.8) GOTO 23
      IF (INPT.GT.8) GOTO 24
   20 CONTINUE
C
C  Input of the structural information.  N=# of modes, NRB=# of rigid
C body modes (set at two for now), NS=# of structural modes,
C NC=# of control surface modes.  OMEGA-modal frequencies
C GM-generalized mass,GK-generalized stiffness.
C
      WRITE(*,'(A)') ' Is the structural information in a file ?'
      READ(*,111) Q1
      IF (Q1.EQ.'Y') THEN
       WRITE(*,'(A)') ' Please input the name of the file.'
       READ(*,111) STRUCT
       OPEN(1,FILE=STRUCT,STATUS='OLD')
       READ(1,116) N,NRB,NS,NC
       N2=2*N
       READ(1,117) (GM(I),I=1,N)
       READ(1,117) (OMEGA(I),I=1,N)
      ELSE
       WRITE(*,'(A)') ' Do you wish to save the data on a file for'
       WRITE(*,'(A)') ' future use?'
       READ(*,111) Q2
       IF (Q2.EQ.'Y') THEN
        WRITE(*,'(A)') ' What is the name of the new file?'
        READ(*,111) STRUCT
        OPEN(1,FILE=STRUCT,STATUS='NEW')
       ENDIF
       WRITE(*,'(A)') ' Input total # of modes used (max 20).'
       READ(*,*) N
       N2=2*N
       WRITE(*,'(A)') ' Input # of rigid modes used (2).'
       READ(*,*) NRB
       WRITE(*,'(A)') ' Input # of structural modes used (max 15).'
       READ(*,*) NS
       WRITE(*,'(A)') ' Input # of control surface modes used (max 5).'
       READ(*,*) NC
       WRITE(*,'(A)') ' Input the generalized masses in the order of'
       WRITE(*,'(A)') ' rigid modes (plunge then pitch), structural'
       WRITE(*,'(A)') ' modes, and control surface modes.'
       READ(*,*) (GM(I),I=1,N)
       WRITE(*,'(A)') ' Input the modal frequencies (in Hz)in the order'
       WRITE(*,'(A)') ' of rigid modes (plunge then pitch), structural '
       WRITE(*,'(A)') ' modes, and control surface modes.'
      READ(*,*) (OMEGA(I),I=1,N)
      IF (Q2.EQ.'Y') THEN
       WRITE(1,116) N,NRB,NS,NC
       WRITE(1,117) (GM(I),I=1,N)
       WRITE(1,117) (OMEGA(I),I=1,N)
      ENDIF
     ENDIF
     GOTO 24
  21 CONTINUE
C
C  Input of the genaralized aerodynamic forces.  It is in the format
C of the output of FASTOP as modified by Max Blair, where the two
C reduced frequencies are combined into one file.
C K1 is the first reduced frequency, K2 is the second reduced frequency.
C GF1 are the generalized forces related to the first reduced frequency,
C GF2 are the generalized forces related to K2. K1 is usually 0.0 and
C K2 is a small reduced frequency (0.001).
C
     WRITE(*,'(A)') ' What is the name of the file for the generalized'
     WRITE(*,'(A)') ' forces (aerodynamics)?'
     READ(*,111) AERO
     OPEN(2,FILE=AERO,STATUS='OLD')
     READ(2,111) TITLE
     READ(2,128) N,XMACH
     READ(2,127) K1
     READ(2,126) (((F(I,J),G(I,J)),I=1,N),J=1,N)
     DO 17 I=1,N
      DO 17 J=1,N
       GF1(I,J)=CMPLX(F(I,J),G(I,J))
  17 CONTINUE
     READ(2,127) K2
     READ(2,126) (((F(I,J),G(I,J)),I=1,N),J=1,N)
     DO 18 I=1,N
      DO 18 J=1,N
       GF2(I,J)=CMPLX(F(I,J),G(I,J))
  18 CONTINUE
   8 CONTINUE
     GOTO 24
  22 CONTINUE
C
C  Input of initial conditions, velocity, density, reference area,
C reference chord (should be the actual A/C area & chord).
C Also the appropriate magnitudes of the mode shapes are input.
C
     WRITE(*,'(A)') ' Are the initial conditions in a file ?'
     READ(*,111) Q3
     IF (Q3.EQ.'Y') THEN
      WRITE(*,'(A)') ' Please input the name of the file.'
      READ(*,111) INCOND
      OPEN(3,FILE=INCOND,STATUS='OLD')
      READ(3,118) RHOO,AO,SREF,CBAR,BR
```

```
        IF (NRB.NE.0) READ(3,119) ALPHAO,HO
        IF (NS.NE.0) READ(3,119) (ZETAO(I),I=1,NS)
        IF (NC.NE.0) READ(3,119) (DELTAO(I),I=1,NC)
       ELSE
        WRITE(*,'(A)') ' Do you wish to save the initial conditions '
        WRITE(*,'(A)') ' on a file for future use?'
        READ(*,111) Q4
        IF (Q4.EQ.'Y') THEN
         WRITE(*,'(A)') ' What is the name of the new file?'
         READ(*,111) INCOND
         OPEN(3,FILE=INCOND,STATUS='NEW')
        ENDIF
        WRITE(*,'(A)') ' Please input the following initial conditions.'
        WRITE(*,'(A)') ' Density (slugs/ft**3):'
        READ(*,*) RHOO
        WRITE(*,'(A)') ' Speed of sound (ft/sec) at density altitude:'
        READ(*,*) AO
        WRITE(*,'(A)') ' Reference Area (in ft**2):'
        READ(*,*) SREF
        WRITE(*,'(A)') ' Reference Chord (MAC) (in ft):'
        READ(*,*) CBAR
        WRITE(*,'(A)') ' What is the reference semichord used in the'
        WRITE(*,'(A)') '  aerodynamics routine (from k=bw/V) in ft.'
        READ(*,*) BR
        IF (NRB.EQ.0) GOTO 19
        WRITE(*,'(A)') ' Pitch mode shape magnitude:'
        READ(*,*) ALPHAO
        WRITE(*,'(A)') ' Plunge mode shape magnitude:'
        READ(*,*) HO
   19   CONTINUE
        IF (NS.EQ.0) GOTO 5
        DO 5 I=1,NS
         WRITE(*,108)I
         READ(*,*) ZETAO(I)
    5   CONTINUE
        IF (NC.EQ.0) GOTO 6
        DO 6 I=1,NC
         WRITE(*,109)I
         READ(*,*) DELTAO(I)
    6   CONTINUE
        IF(Q4.EQ.'Y') THEN
         WRITE(3,118) RHOO,AO,SREF,CBAR,BR
         IF (NRB.NE.0) WRITE(3,119) ALPHAO,HO
         IF (NS.NE.0) WRITE(3,119) (ZETAO(I),I=1,NS)
         IF (NC.NE.0) WRITE(3,119) (DELTAO(I),I=1,NC)
        ENDIF
       ENDIF
       GOTO 24
   26 CONTINUE
C
C  Manual input of velocity root locus data
```

```
C
      WRITE(*,'(A)') ' Do you wish to perform a velocity root locus?'
      READ(*,100) Q5
      IF(Q5.EQ.'Y') THEN
       WRITE(*,'(A)') ' Are the values on file?'
       READ(*,100) Q6
       IF (Q6.EQ.'Y') THEN
        WRITE(*,'(A)') ' What is the name of the file?'
        READ(*,100) RTLCSF
        OPEN(7,FILE=RTLCSF,STATUS='OLD')
        READ(7,121) NVEL
        DO 16 I=1,NVEL
         READ(7,122) V(I)
   16   CONTINUE
       ELSE
        WRITE(*,'(A)') ' Do you want to save this data for future use?'
        READ(*,111) Q7
        IF(Q7.EQ.'Y') THEN
         WRITE(*,'(A)') ' What is the name cf the new file?'
         READ(*,111) RTLCSF
         OPEN(7,FILE=RTLCSF,STATUS='NEW')
        ENDIF
        WRITE(*,'(A)') ' How many velocities tc use (max 20)?'
        READ(*,*) NVEL
        IF (Q7.EQ.'Y') WRITE(7,101) NVEL
        WRITE(*,'(A)') ' Please input the velocities (in ft/sec).'
        READ(*,*) (V(I),I=1,NVEL)
        IF (Q7.EQ.'Y') WRITE(7,102) (V(I),I=1,NVEL)
       ENDIF
      ENDIF
      GOTO 24
   25 CONTINUE
C
C  Manual input of the forcing function matrix
C
      WRITE(*,'(A)') ' Is the forcing function matrix on file?'
      READ(*,111) Q8
      IF(Q8.EQ.'Y') THEN
       WRITE(*,'(A)') ' Please input the name of the file.'
       READ(*,111) INPTM
       OPEN(11,FILE=INPTM,STATUS='OLD')
       READ(11,121) NINP
       READ(11,123) ((QMAT(I,J),J=1,NINP),I=1,N)
      ELSE
       WRITE(*,'(A)') ' Do you wish to save the data to file?'
       READ(*,111) Q9
       IF (Q9.EQ.'Y') THEN
        WRITE(*,'(A)') ' What is the name of the new file?'
        READ(*,111) INPTM
        OPEN(11,FILE=INPTM,STATUS='NEW')
       ENDIF
```


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART

```
      WRITE(*,'(A)') ' How many inputs are there?'
      READ(*,*) NINP
      WRITE(*,124) N,NINP
      DO 30 I=1,N
       WRITE(*,125) I
       READ(*,*) (QMAT(I,J),J=1,NINP)
 30   CONTINUE
      IF (Q9.EQ.'Y') THEN
       WRITE(11,121) NINP
       WRITE(11,123) ((QMAT(I,J),J=1,NINP),I=1,N)
      ENDIF
     ENDIF
     GOTO 24
 27  CONTINUE
C
C  Manual input of the C matrix
C
     WRITE(*,'(A)') ' Is the C matrix on file?'
     READ(*,111) Q10
     IF(Q10.EQ.'Y') THEN
      WRITE(*,'(A)') ' Please input the name of the file.'
      READ(*,111) OUTPM
      OPEN(12,FILE=OUTPM,STATUS='OLD')
      READ(12,121) NOUT
      READ(12,123) ((CMAT(I,J),J=1,N2),I=1,NOUT)
     ELSE
      WRITE(*,'(A)') ' Do you wish to save the data to file?'
      READ(*,111) Q11
      IF (Q11.EQ.'Y') THEN
       WRITE(*,'(A)') ' What is the name of the new file?'
       READ(*,111) OUTPM
       OPEN(12,FILE=OUTPM,STATUS='NEW')
      ENDIF
      WRITE(*,'(A)') ' How many outputs are there?'
      READ(*,*) NOUT
      WRITE(*,129) NOUT,N2
      DO 34 I=1,NOUT
       WRITE(*,125) I
       READ(*,*) (CMAT(I,J),J=1,N2)
 34   CONTINUE
      IF (Q11.EQ.'Y') THEN
       WRITE(12,121) NOUT
       WRITE(12,123) ((CMAT(I,J),J=1,N2),I=1,NOUT)
      ENDIF
     ENDIF
     GOTO 24
 23  CONTINUE
     DO 9 I=1,N
      GK(I)=GM(I)*((OMEGA(I)*2.*PI)**2)
  9  CONTINUE
     VO=XMACH*AO
```

```
      WRITE(4,100) N
      WRITE(4,101) NRB
      WRITE(4,102) NS
      WRITE(4,103) NC
      WRITE(4,104)
      DO 1 I=1,N
       WRITE(4,105) I,I,GM(I)
    1 CONTINUE
      WRITE(4,106)
      DO 2 I=1,N
       WRITE(4,105) I,I,GK(I)
    2 CONTINUE
C
C Call to the routine that non-dimensionalizes FASTOP aero forces
C
      CALL FASTCHG
      IF (NAERO.EQ.0) GOTO 4
      WRITE(4,114) XMACH
      WRITE(4,107) K1
      DO 3 I=1,N
       WRITE(4,125) I
       WRITE(4,112) ((REAL(GF1(I,J)),AIMAG(GF1(I,J))),J=1,N)
    3 CONTINUE
      WRITE(4,107) K2
      DO 4 I=1,N
       WRITE(4,125) I
       WRITE(4,112) ((REAL(GF2(I,J)),AIMAG(GF2(I,J))),J=1,N)
    4 CONTINUE
      WRITE(4,'(/A)') ' The initial conditions are:'
      WRITE(4,115) RHOO,SREF,CBAR,BR
      WRITE(4,'(A)') '1'
  100 FORMAT(/,' The total number of modes are:',I3)
  101 FORMAT(' The total number of rigid body modes are:',I3)
  102 FORMAT(' The total number of elastic modes are:',I3)
  103 FORMAT(' The total number of control surface modes are:',I3)
  104 FORMAT(/,' The generalized mass matrix is',/,' RowCol   Gen Mass')
  105 FORMAT(1X,2I3,E12.4)
  106 FORMAT(/,' The generalized stiffness matrix is',
     0/,' RowCol   Gen Stiff')
  107 FORMAT(/,' The generalized force matrix for reduced frequency of',
     0F7.4,' is')
  108 FORMAT(' Structural mode',I2,' magnitude:')
  109 FORMAT(' Control surface mode',I2,' magnitude:')
  110 FORMAT(1X,I4,1X,I4,2E15.7)
  111 FORMAT(A)
  112 FORMAT(3(1X,2E12.4,'i'))
  113 FORMAT(15X,E15.7)
  114 FORMAT(/,' For a Mach number of',F6.3)
  115 FORMAT(' Density=',F11.5,'Slugs/ft**3, Reference Area=',
     0E11.4,' ft**2',/,
     0' Chord=',E11.4,' feet, and an aerodynamic reference',
   0' semi-chord=',E11.4,' feet.')
116 FORMAT(4I5)
117 FORMAT(4E15.7)
118 FORMAT(F10.7,F10.2,3F10.5)
119 FORMAT(6F10.5)
120 FORMAT(' MAC ready for input',//,
   0'  Enter (#): ',//,
   0'      1.  Automatic input of items (2)-(7)',/
   0'      2.  Read in structural data',/
   0'      3.  Read in aerodynamic data',/
   0'      4.  Read in initial conditions',/
   0'      5.  Read in root locus values',/
   0'      6.  Read in forcing function matrix',/
   0'      7.  Read in the C matrix',/
   0'      8.  Completion of input data',//)
121 FORMAT(I5)
122 FORMAT(F10.3)
123 FORMAT(6E15.7)
124 FORMAT( ' For n modes, there are n modes with m controls',/,
   0' The modes will be set up as follows:',/,
   0'   Rigid modes',/,
   0'   Structural modes',/,
   0'   Control surface modes',/,
   0' The matrix will have',I3,' rows and ',I3,' columns',/,
   0' Please input the matrix by row.'//)
125 FORMAT(' Row',I3)
126 FORMAT(8E15.7)
127 FORMAT(E15.7)
128 FORMAT(I5,5X,E15.7)
129 FORMAT(' The matrix will have ',I3,' rows and',I3,' columns',/
   0' Please input the matrix by row.'//)
    RETURN
    END
```

```
      SUBROUTINE MATSAV ( LUNIT, NAME, NR, M, N, IMG,
     $                    XREAL, XIMAG, FORMT )
C
C ------------------------------------------------------------------
C |                                                                |
C |                              (TM)                             |
C |                        MATRIX      V5.0                        |
C |                              X                                 |
C |                                                                |
C |          (C) COPYRIGHT INTEGRATED SYSTEMS, INC. 1985           |
C |                    PALO ALTO, CALIFORNIA                       |
C |                                                                |
C |                     All Rights Reserved                        |
C |                                                                |
C |          This software may not be copied or altered            |
C |          without  the  express  written  consent of            |
C |                  Integrated Systems, Inc.                      |
C |                                                                |
C |----------------------------------------------------------------|
C |                                                                |
C | MATSAV writes a matrix to a file in a format suitable for the  |
C |       MATRIXx LOAD operation.                                  |
C |                                                                |
C |----------------------------------------------------------------|
C | Param. | Type            | On input-                | On output- |
C |----------------------------------------------------------------|
C |        |                 |                          |            |
C | LUNIT  | INTEGER         | Fortran logical unit number. | unchanged. |
C |        |                 |                          |            |
C | NAME   | CHARACTER*(*)   | Name of the matrix.  One al- | unchanged. |
C |        | (maximum        | phabetic followed by up to 9 |            |
C |        |  length 10)     | alphanumeric characters. |            |
C |        |                 |                          |            |
C | NR     | INTEGER         | Row-dimension in the     | unchanged. |
C |        |                 | defining dimension or type |          |
C |        |                 | statement in the calling |            |
C |        |                 | program. NR must be greater |         |
C |        |                 | than or equal to M.      |            |
C |        |                 |                          |            |
C | M      | INTEGER         | Number of rows of the matrix | unchanged. |
C |        |                 |                          |            |
C | N      | INTEGER         | Number of columns of the | unchanged. |
C |        |                 | matrix.                  |            |
C |        |                 |                          |            |
C | IMG    | INTEGER         | If IMG = 0, the imaginary | unchanged. |
C |        |                 | part (XIMAG) is assumed to |          |
C |        |                 | be zero and is not saved. |           |
C |        |                 |                          |            |
C | XREAL  | DOUBLE          | Real part of the matrix to | unchanged. |
C |          | PRECISION      | be saved.                      |             |
C |          |                |                                |             |
C | XIMAG    | DOUBLE         | Imaginary part of the matrix   | unchanged.  |
C |          | PRECISION      | to be saved.                   |             |
C |          |                |                                |             |
C | FORMT    | CHARACTER*(*)  | String containing the For-     | unchanged.  |
C |          | (maximum       | tran format to be used for     |             |
C |          |  length 20)    | writing the elements of the    |             |
C |          |                | matrix.                        |             |
C |          |                |                                |             |
C |--------------------------------------------------------------------------|
C |                                                                          |
C | Example:  The following Fortran program generates an elementary          |
C |           matrix in X and writes it to Fortran unit 1.  Assume           |
C |           that unit 1 has been preallocated as file (data set)           |
C |           TEST.                                                          |
C |                                                                          |
C |                                                                          |
C |                DOUBLE PRECISION X(20,3), DUMMY                           |
C |                  DO 200 J=1,3                                            |
C |                    DO 100 I=1,10                                         |
C |                    X(I,J)=0.0D0                                          |
C |           100      CONTINUE                                              |
C |                  X(J,J)=1.0D0                                            |
C |           200    CONTINUE                                                |
C |                CALL MATSAV( 1, 'AMATRIX', 20, 10, 3, 0,                  |
C |               $             X, DUMMY, '(1P2E24.15)' )                    |
C |                STOP                                                      |
C |                END                                                       |
C |                                                                          |
C |                                                                          |
C | After this program runs, invoke MATRIXx and type:                        |
C |                                                                          |
C |           <> LOAD 'TEST'                                                 |
C |                                                                          |
C | This will put X on the stack as stack-variable-name AMATRIX.             |
C |                                                                          |
C ----------------------------------------------------------------------------
C
      INTEGER             LUNIT, M, N, NR, IMG
      CHARACTER*(*)       NAME, FORMT
      DOUBLE PRECISION    XREAL(NR,1), XIMAG(NR,1)
      CHARACTER           NAM*10, FORM*20
C
C                                ---------------------------------------
C                                | write header record.                 |
C                                ---------------------------------------
      NAM=NAME
      FORM=FORMT
      WRITE(LUNIT,'(A10,3I5,A20)') NAM,M,N,IMG,FORM
C                                ---------------------------------------
```

```
C                                  | write real-part of the matrix.         |
C                                  -----------------------------------------
      WRITE(LUNIT,FORM) ((XREAL(I,J),I=1,M),J=1,N)
C                                  -----------------------------------------
C                                  | write imaginary-part if nonzero.       |
C                                  -----------------------------------------
      IF(IMG.NE.0) WRITE(LUNIT,FORM) ((XIMAG(I,J),I=1,M),J=1,N)
      RETURN
      END
      SUBROUTINE MINV
C
C  This subroutine inverts the X matrix using the IMSL routine
C LINV1F.
C
      REAL X,XINV,WKAREA
      DIMENSION X(20,20),XINV(20,20),WKAREA(20)
      COMMON/INV/X,XINV
      INTEGER N,NRB,NS,NC,IER
      COMMON/INDEX/N,NRB,NS,NC
      INTEGER NAERO,NMAT,NINV,NAMAT,NEIG
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG
C
C  Call to IMSL matrix inversion routine
C
      CALL LINV1F(X,N,20,XINV,4,WKAREA,IER)
      IF (IER.EQ.34) THEN
       WRITE(*,403)
       GOTO 5
      ENDIF
      IF (IER.NE.0) THEN
       WRITE(*,400) IER
       STOP
      ENDIF
    5 CONTINUE
      IF(NINV.EQ.0) GOTO 2
      WRITE(4,'(/A)') ' X inverse is'
      DO 2 I=1,N
       WRITE(4,401) I
       DO 3 J=1,N,6
        WRITE(4,402) (XINV(I,K),K=J,J+5)
    3  CONTINUE
    2 CONTINUE
  400 FORMAT(///,' **** There was an error in inverting X, # ',I3,
     0' ****',///)
  401 FORMAT(' ROW',I3)
  402 FORMAT(6E12.4)
  403 FORMAT(///,' The inversion did not meet error criterion',///)
    1 CONTINUE
      RETURN
      END
      SUBROUTINE MMULT
C
C  This subroutine multiplies Xinv*Y and Xinv*Z using the
C IMSL routine VMULFF
C
      REAL Y,Z,XINV,XINVY,XINVZ
      DIMENSION X(20,20),Y(20,20),Z(20,20),XINV(20,20),
     @XINVY(20,20),XINVZ(20,20)
      COMMON/MMUL/Y,Z,XINVY,XINVZ
      COMMON/INV/X,XINV
      INTEGER N,NRB,NS,NC,IER
      COMMON/INDEX/N,NRB,NS,NC
      INTEGER NAERO,NMAT,NINV,NAMAT,NEIG
      COMMON/PRT/NAERO,NMAT,NINV,NAMAT,NEIG
C
C  Call to IMSL matrix multiplication routine to multiply
C XINV*Y
C
      CALL VMULFF(XINV,Y,N,N,N,20,20,XINVY,20,IER)
      IF (IER.NE.0) THEN
       WRITE(*,500) IER
       STOP
      ENDIF
      IF(NINV.EQ.0) GOTO 2
      WRITE(4,'(/A)') ' XINV*Y is'
      DO 2 I=1,N
       WRITE(4,501) I
       DO 3 J=1,N,6
        WRITE(4,502) (XINVY(I,K),K=J,J+5)
    3  CONTINUE
    2 CONTINUE
      IER=0
C
C  Call to IMSL matrix multiplication routine to multiply
C XINV*Z
C
      CALL VMULFF(XINV,Z,N,N,N,20,20,XINVZ,20,IER)
      IF (IER.NE.0) THEN
       WRITE(*,500) IER
       GOTO 1
      ENDIF
      IF(NINV.EQ.0) GOTO 4
      WRITE(4,'(/A)') ' XINV*Z is'
      DO 4 I=1,N
       WRITE(4,501) I
       DO 5 J=1,N,6
        WRITE(4,502) (XINVZ(I,K),K=J,J+5)
    5  CONTINUE
    4 CONTINUE
  500 FORMAT(///,' ****There was an error in the matrix multiply, # '
    0,I3,' ****',///)
501 FORMAT(' ROW',I3)
502 FORMAT(6E12.4)
  1 CONTINUE
    RETURN
    END
```

```
      SUBROUTINE RLPLOT(FL,FH,RL,RH,TITLE)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XPOS,YPOS
      CHARACTER*4 NUM
      CHARACTER*23 DT
      CHARACTER*40 TITLE
      LOGICAL PLOT
      REAL VAL(13)/.01,.1,.2,.5,1.,2.,5.,10.,20.,50.,100.,200.,500./
C
      REWIND 90
      ZERO=0.DO
      ONE=1.DO
       WRITE(*,1010)
       READ(*,*)
      CALL JBEGIN
C CLEAR SCREEN WITH PLOT10 COMMANDS
      CALL INITT(960)
      CALL FINITT(0,2800)
C
C SETUP GRAPHICS
C
       PLOT=.FALSE.
30      IF(.NOT. PLOT) THEN
        CALL JDINIT(1)
        CALL JDEVON(1)
       ELSE
        CALL JDINIT(2)
        CALL JDEVON(2)
       END IF
C
       CALL JWINDO(-4.,12.,-3.,15.)
       CALL JOPEN
       CALL JSIZE(.27,.35)
       IF(PLOT) CALL JFONT(5)
       IF(.NOT.PLOT) CALL JFONT(1)
C
C  DRAW BOX AND SET UP AXES
C
       CALL JMOVE(0.,0.)
       CALL JDRAW(10.,0.)
       CALL JDRAW(10.,10.)
       CALL JDRAW(0.,10.)
       CALL JDRAW(0.,0.)
C
       XRANGE=RH-RL
       XRAT=XRANGE/10.
       YRANGE=FH-FL
       YRAT=YRANGE/10.
C
       DO 50 I=1,13
```

```
        IF(XRANGE.GE.VAL(I)) XSTEP=VAL(I)/5.
50      IF(YRANGE.GE.VAL(I)) YSTEP=VAL(I)/5.
C
       XPOS=SNGL(-RL/XRAT)
       CALL JMOVE(XPOS,0.)
       CALL JDRAW(XPOS,10.)
C
C  DRAW TIC MARKS AND NUMBERS
C
       X=RL
60     XPOS=SNGL(X/XRAT-RL/XRAT)
       IF (XPOS.GT.10.)GOTO 65
       CALL JMOVE(XPOS,0.)
       CALL JRDRAW(0.,-.3)
C
       ENCODE(4,1020,NUM)X
       CALL JJUST(2,3)
       CALL JMOVE(XPOS,-.5)
       CALL JHSTRG(NUM)
C
       X=X+XSTEP
       GOTO 60
C
65     Y=FL
70     YPOS=SNGL(Y/YRAT-FL/YRAT)
       IF(YPOS.GT.10.) GOTO 75
       ENCODE(4,1020,NUM)Y
        CALL JJUST(3,2)
        CALL JMOVE(0.,YPOS)
        CALL JRDRAW(-.3,0.)
        CALL JMOVE(-.5,YPOS)
        CALL JHSTRG(NUM)
       Y=Y+YSTEP
       GOTO 70
C
75      CALL JSIZE(.4,.45)
        CALL JJUST(2,3)
        CALL JBASE(1.,0.,0.)
        CALL JMOVE(5.,-2.)
       CALL JHTEXT(42,42HR[BLC]EAL [ELC]C[BLC]OMPONENT (1/SEC)[ELC])
        CALL JBASE(0.,1.,0.)
        CALL JJUST(2,1)
        CALL JMOVE(-2.,5.)
       CALL JHTEXT(44,44HI[BLC]MAGINARY [ELC]C[BLC]OMPONENT (HZ)[ELC])
C
C PRINT TITLE AND DATE AT TOP OF PLOT
C
C        CALL JBASE(1.,0.,0.)
C        CALL JJUST(1,1)
C        CALL JMOVE(-3.,12.)
```

```
       CALL JHSTRG(TITLE)
C       CALL LIB$DATE_TIME(DT)
C       CALL JMOVE(-3.,11.)
C       CALL JHSTRG(DT)
C
80     READ(90,END=999) XH,YH,RF
       IF(XH.GT.RH.OR.XH.LT.RL) GOTO 80
       IF(YH.GT.FH.OR.YH.LT.FL) GOTO 80
       XPOS=SNGL(XH/XRAT-RL/XRAT)
       YPOS=SNGL(YH/YRAT-FL/YRAT)
      CALL JMOVE(XPOS,YPOS)
C
C  DRAW A LITTLE SQUARE
C
       CALL JRDRAW(.05,.05)
       CALL JRDRAW(-.1,0.)
       CALL JRDRAW(0.,-.1)
       CALL JRDRAW(.1,0.)
       CALL JRDRAW(0.,.1)
       GOTO 80
C
C SHUT EVERYTHING OFF
C
999   CALL JCLOSE
      IF(.NOT.PLOT) THEN
       CALL JPAUSE(1)
       CALL JDEVOF(1)
       CALL JDEND(1)
      ELSE
       CALL JPAUSE(2)
       CALL JDEVOF(2)
       CALL JDEND(2)
      END IF
C
C SEND QMS FILE TO THE PRINTER
C
C      IF(PLOT)THEN
C       WRITE(*,1040)
C       ISPAWN=LIB$SPAWN('Q[MAX.ASE.ADAM.DI3SPAWN]MAXRL.COM')
C       IF(.NOT.ISPAWN) CALL LIB$SIGNAL(%VAL(ISPAWN))
C      PLOT=.FALSE.
C      END IF
C
C MENU
C CLEAR SCREEN WITH PLOT10 COMMANDS
210   CALL INITT(960)
      CALL FINITT(0,2800)
      WRITE(*,1030)
      READ(*,*)IGOTO
       IF(IGOTO.LT.1.OR.IGOTO.GT.3) GOTO 210
      GOTO (300,400,500),IGOTO
```

```
C
C RETURN TO ADAM
300     CALL JEND
        WRITE(*,1050)
        READ(*,*)
        RETURN
C
C CHANGE PLOT LIMITS
C
400     WRITE(*,1060) RL,RH,FL,FH
        WRITE(*,1070)
        READ(*,1065)RL
        WRITE(*,1080)
        READ(*,1065)RH
        WRITE(*,1090)
        READ(*,1065)FL
        WRITE(*,1100)
        READ(*,1065)FH
        REWIND 90
        GOTO 30
C
C MAKE HARDCOPY OF CURRENT PLOT
500     PLOT=.TRUE.
        REWIND 90
        GOTO 30
C
1010    FORMAT(' SWITCH TO GRAPHICS MODE AND HIT RETURN')
1020    FORMAT(F4.0)
1030    FORMAT(///' PLOT ROUTINE MENU'//'  1-RETURN TO MAC'//'  2-CHANGE
     1  PLOT LIMITS'//'  3-MAKE A HARDCOPY OF CURRENT PLOT'//)
1040    FORMAT(' SUBPROCES SPAWNED.  FILE ON ITS WAY TO PLOTTER')
1050    FORMAT(' RETURN TERMINAL TO STANDARD MODE AND HIT RETURN KEY')
1060    FORMAT(' XMIN=',F8.4/' XMAX=',F8.4/' YMIN=',F8.4/' YMAX=',F8.4)
1065    FORMAT(F9.5)
1070    FORMAT(' NEW XMIN ?'/)
1080    FORMAT(' NEW XMAX ?'/)
1090    FORMAT(' NEW YMIN ?'/)
1100    FORMAT(' NEW YMAX ?'/)
1110    FORMAT(' NEW ROW ?'/)
1120    FORMAT(' NEW COL ?'/)
1130    FORMAT(I3)
        END
```

```
      SUBROUTINE STABDER(NVEL,V)
C
C  This subroutine computes the dimensional, and some non-dimensional
C stability derivatives for use in developing the reduced order model.
C
      CHARACTER*1 QCHANGE
      INTEGER N,NRB,NS,NC
      COMMON/INDEX/N,NRB,NS,NC
      REAL K1,K2
      COMPLEX GF1,GF2
      DIMENSION GF1(20,20),GF2(20,20)
      COMMON/AERO/K1,K2,GF1,GF2,PI
      REAL VO,RHOO,SREF,CBAR,ALPHAO,HO,ZETAO,DELTAO,BR
      DIMENSION ZETAO(15),DELTAO(5)
      COMMON/IC/VO,RHOO,SREF,CBAR,ALPHAO,HO,ZETAO,DELTAO,BR
      REAL ZALPHA,MALPHA,ZALPHDT,MALPHDT,ZQ,MQ,
     @FZALPHA,FZALPDTN,FZETQ,ZZETA,MZETA,
     @ZZETADT,MZETADT,ZZETDDT,MZETDDT,
     @FZETZET,FZETZDT,FZEZDDT,
     @FZETDEL,FZETDDT,FZEDDDT,
     @FDELZET,FDELZDT,FDEZDDT,
     @ZDELTA,MDELTA,FDALPHA,FDALPDT,FDETQ,
     @ZDELTDT,MDELTDT,ZDELDDT,MDELDDT,
     @FDELDEL,FDELDDT,FDEDDDT,VEL
      DIMENSION FZALPHA(15),FZALPDT(15),FZETQ(15),ZZETA(15),MZETA(15),
     @ZZETADT(15),MZETADT(15),ZZETDDT(15),MZETDDT(15),
     @FZETZET(15,15),FZETZDT(15,15),FZEZDDT(15,15),
     @FZETDEL(15,5),FZETDDT(15,5),FZEDDDT(15,5),
     @FDELZET(5,15),FDELZDT(5,15),FDEZDDT(5,15),
     @ZDELTA(5),MDELTA(5),FDALPHA(5),FDALPDT(5),FDETQ(5),
     @ZDELTDT(5),MDELTDT(5),ZDELDDT(5),MDELDDT(5),
     @FDELDEL(5,5),FDELDDT(5,5),FDEDDDT(5,5)
      COMMON/DERIV/ZALPHA,MALPHA,ZALPHDT,MALPHDT,ZQ,MQ,FZALPHA,FZALPDT,
     @FZETQ,ZZETA,MZETA,ZZETADT,MZETADT,ZZETDDT,MZETDDT,FZETZET,
     @FZETZDT,FZEZDDT,FZETDEL,FZETDDT,FZEDDDT,FDELZET,FDELZDT,FDEZDDT,
     @ZDELTA,MDELTA,FDALPHA,FDALPDT,FDETQ,
     @ZDELTDT,MDELTDT,ZDELDDT,MDELDDT,FDELDEL,FDELDDT,FDEDDDT,VEL
      REAL CMALPHA,CZALPHA,CMALPDT,CZALPDT,CMQ,CZQ,CMDELTA,
     @CZDELTA,MODE
      DIMENSION MODE(20),CMDELTA(5),CZDELTA(5)
C
C  If a velocity root locus is desired, then the velocity takes on
C the velocities desired, otherwise the initial velocity is used.
C
      IF (NVEL.EQ.0) THEN
       VEL=VO
      ELSE
       VEL=V
      ENDIF
      WRITE(*,203) VEL,RHOO
      WRITE(4,203) VEL,RHOO
      QBAR=0.5*RHOO*VEL**2
      IF (VEL.EQ.0.0) THEN
       QBAR=0.0
       VEL=0.001
      ENDIF
      IF (NRB.EQ.0) GOTO 29
      MODE(1)=HO
      MODE(2)=ALPHAO
   29 CONTINUE
      IF (NS.EQ.0) GOTO 1
      DO 1 I=1,NS
       J=I+NRB
       MODE(J)=ZETAO(I)
    1 CONTINUE
      IF (NC.EQ.0) GOTO 2
      DO 2 I=1,NC
       J=I+NRB+NS
       MODE(J)=DELTAO(I)
    2 CONTINUE
C
C  Computation of all the dimensional derivatives is accomplished
C below.  They are based on the paper by Rodden, AIAA
C and the formula for the generalized forces from FASTOP
C
      IF(NRB.EQ.0) GOTO 20
      ZALPHA=REAL(GF1(1,2))*QBAR*SREF/(MODE(1)*MODE(2))
      MALPHA=REAL(GF1(2,2))*QBAR*SREF*CBAR/(MODE(2)**2)
      ZALPHDT=-REAL(GF2(1,1))*(SREF*QBAR*BR**2/(K2**2*VEL))/
     0(MODE(1)**2)
      MALPHDT=-REAL(GF2(2,1))*(SREF*CBAR*QBAR*BR**2/(K2**2*VEL))/
     0(MODE(1)*MODE(2))
      ZQ=AIMAG(GF2(1,2))*(SREF*QBAR*BR/(K2*VEL))/(MODE(1)*MODE(2))-
     0ZALPHDT
      MQ=AIMAG(GF2(2,2))*(SREF*CBAR*QBAR*BR/(K2*VEL))/(MODE(2)**2)-
     0MALPHDT
   20 CONTINUE
      IF (NS.EQ.0) GOTO 22
      DO 3 I=1,NS
       J=I+NRB
       IF (NRB.EQ.0) GOTO 21
       FZALPHA(I)=REAL(GF1(J,2))*QBAR*SREF/(MODE(J)*MODE(2))
       FZALPDT(I)=-REAL(GF2(J,1))*(SREF*QBAR*BR**2/(K2**2*VEL))/
     0(MODE(1)*MODE(J))
       FZETQ(I)=AIMAG(GF2(J,2))*(SREF*QBAR*BR/(K2*VEL))/(MODE(J)*
     0MODE(2))-FZALPDT(I)
       ZZETA(I)=REAL(GF1(1,J))*QBAR*SREF/(MODE(1)*MODE(J))
       MZETA(I)=REAL(GF1(2,J))*QBAR*SREF*CBAR/(MODE(2)*MODE(J))
       ZZETADT(I)=AIMAG(GF2(1,J))*(SREF*QBAR*BR/(K2*VEL))/
     0(MODE(1)*MODE(J))
       MZETADT(I)=AIMAG(GF2(2,J))*(SREF*CBAR*QBAR*BR/(K2*VEL))/
     0(MODE(2)*MODE(J))
       ZZETDDT(I)=-(REAL(GF2(1,J))*(SREF*QBAR/(MODE(1)*MODE(J)))
     0-ZZETA(I))/((K2*VEL/BR)**2)
       MZETDDT(I)=-(REAL(GF2(2,J))*(SREF*CBAR*QBAR/
     0(MODE(2)*MODE(J)))-MZETA(I))/((K2*VEL/BR)**2)
 21    CONTINUE
       DO 4 K=1,NS
        L=K+NRB
        FZETZET(I,K)=REAL(GF1(J,L))*QBAR*SREF/(MODE(J)*MODE(L))
        FZETZDT(I,K)=AIMAG(GF2(J,L))*(SREF*QBAR*BR/(K2*VEL))/
     0(MODE(J)*MODE(L))
        FZEZDDT(I,K)=-(REAL(GF2(J,L))*(SREF*QBAR/(MODE(J)*MODE(L)))-
     0FZETZET(I,K))/((K2*VEL/BR)**2)
  4    CONTINUE
       IF (NC.EQ.0) GOTO 5
       DO 5 K=1,NC
        L=K+NRB+NS
        FDELZET(K,I)=RFAL(GF1(L,J))*QBAR*SREF*CBAR/(MODE(L)*MODE(J))
        FZETDEL(I,K)=REAL(GF1(J,L))*QBAR*SREF/(MODE(J)*MODE(L))
        FDELZDT(K,I)=AIMAG(GF2(L,J))*(SREF*CBAR*QBAR*BR/(K2*VEL))/
     0(MODE(L)*MODE(J))
        FZETDDT(I,K)=AIMAG(GF2(J,L))*(SREF*QBAR*BR/(K2*VEL))/
     0(MODE(J)*MODE(L))
        FDEZDDT(K,I)=-(REAL(GF2(L,J))*(SREF*CBAR*QBAR/(MODE(L)*
     0MODE(J)))-FDELZET(K,I))/((K2*VEL/BR)**2)
        FZEDDDT(I,K)=-(REAL(GF2(J,L))*(SREF*QBAR/(MODE(J)*MODE(L)))-
     0FZETDEL(I,K))/((K2*VEL/BR)**2)
  5    CONTINUE
  3   CONTINUE
 22   CONTINUE
      IF (NC.EQ.0) GOTO 23
      DO 6 I=1,NC
       J=I+NRB+NS
       IF (NRB.EQ.0) GOTO 25
       ZDELTA(I)=REAL(GF1(1,J))*QBAR*SREF/(MODE(1)*MODE(J))
       MDELTA(I)=REAL(GF1(2,J))*QBAR*SREF*CBAR/(MODE(2)*MODE(J))
       FDALPHA(I)=REAL(GF1(J,2))*QBAR*SREF*CBAR/(MODE(J)*MODE(2))
       FDALPDT(I)=-REAL(GF2(J,1))*(SREF*CBAR*QBAR*BR**2/
     0(K2**2*VEL))/(MODE(1)*MODE(J))
       FDETQ(I)=AIMAG(GF2(J,2))*(SREF*CBAR*QBAR*BR/(K2*VEL))/
     0(MODE(J)*MODE(2))-FZALPDT(I)
       ZDELTDT(I)=AIMAG(GF2(1,J))*(SREF*QBAR*BR/(K2*VEL))/
     0(MODE(1)*MODE(J))
       MDELTDT(I)=AIMAG(GF2(2,J))*(SREF*CBAR*QBAR*BR/(K2*VEL))/
     0(MODE(2)*MODE(J))
       ZDELDDT(I)=-(REAL(GF2(1,J))*(SREF*QBAR/(MODE(1)*MODE(J)))-
     0ZDELTA(I))/((K2*VEL/BR)**2)
       MDELDDT(I)=-(REAL(GF2(2,J))*(SREF*CBAR*QBAR/
     0(MODE(2)*MODE(J)))-MDELTA(I))/((K2*VEL/BR)**2)
 25    CONTINUE
       IF (NC.EQ.0) GOTO 24
```

```
          DO 6 K=1,NC
           L=K+2+NS
           FDELDEL(I,K)=REAL(GF1(J,L))*QBAR*SREF*CBAR/(MODE(J)*MODE(L))
           FDELDDT(I,K)=AIMAG(GF2(J,L))*(SREF*CBAR*QBAR*BR/(K2*VEL))/
     @(MODE(J)*MODE(L))
           FDEDDDT(I,K)=-(REAL(GF2(J,L))*(SREF*CBAR*QBAR/
     @(MODE(J)*MODE(L)))-FDELDEL(I,K))/((K2*VEL/BR)**2)
   24 CONTINUE
    6 CONTINUE
   23 CONTINUE
C
C  Computation of the non-dimensional stability derivatives of interest
C from the dimensional derivatives.  These can be used for comparison
C against another aerodynamic code to check for accuracy.
C
      IF (NRB.EQ.0) GOTO 26
      CMALPHA=MALPHA/(0.5*RHOO*VEL**2*SREF*CBAR)
      CZALPHA=ZALPHA/(0.5*RHOO*VEL**2*SREF)
      CMALPDT=MALPHDT/(0.5*RHOO*VEL**2*SREF*CBAR)*(2*VEL/CBAR)
      CZALPDT=ZALPHDT/(0.5*RHOO*VEL**2*SREF)*(2*VEL/CBAR)
      CMQ=MQ/(0.5*RHOO*VEL**2*SREF*CBAR)*(2*VEL/CBAR)
      CZQ=ZQ/(0.5*RHOO*VEL**2*SREF)*(2*VEL/CBAR)
      WRITE(*,200) CMALPHA,CMALPDT,CMQ
      WRITE(*,201) CZALPHA,CZALPDT,CZQ
      WRITE(*,'(A)') ' The control surface derivatives are:'
      WRITE(4,200) CMALPHA,CMALPDT,CMQ
      WRITE(4,201) CZALPHA,CZALPDT,CZQ
      WRITE(4,'(A)') ' The control surface derivatives are:'
   26 CONTINUE
      IF (NC.EQ.0) GOTO 7
      DO 7 I=1,NC
       CMDELTA(I)=MDELTA(I)/(0.5*RHOO*VEL**2*SREF*CBAR)
       CZDELTA(I)=ZDELTA(I)/(0.5*RHOO*VEL**2*SREF)
       WRITE(*,202) I,CMDELTA(I),I,CZDELTA(I)
       WRITE(4,202) I,CMDELTA(I),I,CZDELTA(I)
    7 CONTINUE
      IF(NSTAB.EQ.0) GOTO 8
      WRITE(*,'(/A)') ' Do you want to change any of the derivatives?'
      READ(*,204) QCHANGE
      IF(QCHANGE.NE.'Y') GOTO 8
       IF (NRB.EQ.0) GOTO 27
       WRITE(*,'(A)') ' CMalpha:'
       READ(*,*) CMALPHA
       MALPHA=CMALPHA*QBAR*SREF*CBAR
       WRITE(*,'(A)') ' CZalpha:'
       READ(*,*) CZALPHA
       ZALPHA=CZALPHA*QBAR*SREF
       WRITE(*,'(A)') ' CMq:'
       READ(*,*) CMQ
       MQ=CMQ*QBAR*SREF*CBAR**2/(2*VEL)
       WRITE(*,'(A)') ' CZq:'
      READ(*,*) CZQ
      ZQ=CZQ*SREF*CBAR/(2*VEL)
      WRITE(*,'(A)') ' CMalphadt:'
      READ(*,*) CMALPDT
      MALPHDT=CMALPDT*QBAR*SREF*CBAR**2/(2*VEL)
      WRITE(*,'(A)') ' CZalphadt:'
      READ(*,*) CZALPDT
      ZALPHDT=CZALPDT*QBAR*SREF*CBAR/(2*VEL)
 27   CONTINUE
      IF (NC.EQ.0) GOTO 9
      DO 9 I=1,NC
       WRITE(*,205) I
       READ(*,*) CMDELTA(I)
       MDELTA(I)=CMDELTA(I)*QBAR*SREF*CBAR
       WRITE(*,206) I
       READ(*,*) CZDELTA(I)
       ZDELTA(I)=CZDELTA(I)*QBAR*SREF*CBAR
  9   CONTINUE
      IF (NRB.EQ.0) GOTO 28
      WRITE(*,'(/A)') ' The new derivatives are:'
      WRITE(*,200) CMALPHA,CMALPDT,CMQ
      WRITE(*,201) CZALPHA,CZALPDT,CZQ
      WRITE(*,'(A)') ' The new control surface derivatives are:'
      WRITE(4,'(/A)') ' The new derivatives are:'
      WRITE(4,200) CMALPHA,CMALPDT,CMQ
      WRITE(4,201) CZALPHA,CZALPDT,CZQ
      WRITE(4,'(A)') ' The new control surface derivatives are:'
 28   CONTINUE
      IF (NC.EQ.0) GOTO 10
      DO 10 I=1,NC
       CMDELTA(I)=MDELTA(I)/(0.5*RHOO*VEL**2*SREF*CBAR)
       CZDELTA(I)=ZDELTA(I)/(0.5*RHOO*VEL**2*SREF)
       WRITE(*,202) I,CMDELTA(I),I,CZDELTA(I)
       WRITE(4,202) I,CMDELTA(I),I,CZDELTA(I)
 10   CONTINUE
  8   CONTINUE
200   FORMAT(' CMalpha=',E11.4,' CMalphadot=',E11.4,' CMq=',E11.4)
201   FORMAT(' CZalpha=',E11.4,' CZalphadot=',E11.4,' CZq=',E11.4)
202   FORMAT(' CMdelta(',I1,')=',E11.4,' CZdelta(',I1,')=',E11.4)
203   FORMAT(//,' For a velocity of',F10.4,' ft/sec, and a density of',
     0F8.5,'slugs/ft**3')
204   FORMAT(A)
205   FORMAT(' CMdelta(',I1,'):')
206   FORMAT(' CZdelta(',I1,'):')
      RETURN
      END
```

Bibliography

1. Bisplinghoff, Raymond and Holt Ashley. Principles of Aeroelasticity. New York: Dover Publications, 1962.

2. Blair, Maxwell and Thomas Noll. Program ADAM, AFWAL-TR-86-xxxx, Volumes 1 and 2. Air Force Wright Aeronautical Laboratories, Wright-Patterson AFB OH, January 1986.

3. Cavin, R. K., III and A. R. Dusto. "Hamilton's Principle: Finite-Element Methods and Flexible Body Dynamics," AIAA Journal, 15: 1684-1690 (December 1977).

4. Cutchins, Malcolm A. and James W. Purvis. Aeroservoelastitity in the Time Domain, AFATL-TR-79-85. Air Force Armament Laboratory, Eglin AFB FL, October 1979 (AD-B043389).

5. Dusto, A. R. et al. A Method for Predicting the Stability Characteristics of Control Configured Vehicles, Volume 1, FLEXSTAB 2.01.00 Theoretical Description, AFFDL-TR-74-91. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, November 1974.

6. Etkin, Bernard. Dynamics of Atmospheric Flight. New York: Wiley & Sons, 1972.

7. Felt, L. R. et al. "Aeroservoelastic Encounters," AIAA Journal of Aircraft, 16: 477-483 (July 1979).

8. Giesing, J. P. et al. Subsonic Unsteady Aerodynamics for General Configurations, AFFDL-TR-71-5, Part I, Volume I and II. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, November 1971 (AD-891403).

9. Kane, T. R. Dynamics. New York: Holt, Rinehart and Winston, 1968.

10. MATRIXx User's Manual. Integrated Systems Inc., Palo Alto CA, 1984.

11. McDonough, Thomas B. "Formulation of the Global Equations of Motion of a Deformable Body," AIAA Journal, 14: 656-660 (May 1976).

12. Meirovitch, Leonard. Elements of Vibration Analysis. New York: McGraw-Hill Book Copmpany, 1975.

13. Milne, R. D. Dynamics of the Deformable Aeroplane, R & M Number 3345. Her Majesty's Stationary Office, London, England, 1964 (AD-A953155).

14. Noll, Thomas E. et al. "ADAM, an Aeroservoelastic Analysis Method for Analog or Digital Systems," AIAA Paper 85-3090, October 1985.

15. Noll, Thomas E. and Lawrence J. Huttsell. YF-16 Stability Derivatives for Control System/Airframe Interaction Analyses, AFFDL-TM-76-107-FBR. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, December 1976.

16. Noll, Thomas E. Unpublished YF-17 data. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, 1976.

17. Pasquini, Glenn J. Active Suppression of Aeroelastic Instability on a Forward Swept Wing Using a Linear Optimal Regulator. MS thesis, AFIT/GAE/AA/84J-01. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 1984.

18. Proceedings of the Aeroservoelastic Specialists Meeting, AFWAL-TR-84-3105, Volumes 1 and II. Air Force Wright Aeronautical Laboratories, Wright-Patterson AFB OH, October 1984.

19. Rodden, William P. et al. Aero-servo-elastic Stability Analysis, WPR (NASC) TR-79-1. Naval Air Systems Command, Washington DC, April 1979 (AD-A072797).

20. Rodden, William P. and Joseph P. Giesing. "Application of Oscillatory Aerodynamic Theory to Estimation of Dynamic Stability Derivatives," AIAA Journal of Aircraft, 7: 272-275 (May-June 1970).

21. Schwanz, Robert C. et al. "Dynamic Modeling Uncertainty," AIAA Paper 84-1057CP, May 1984.

22. Schwanz, Robert C. A Lagrangian Formulation of Flight Vehicle Dynamics Described by Non-inertial Motion States, AFFDL-TM-78-83-FGC. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, August 1978.

23. Schwanz, Robert C. Formulations of the Equations of Motion of an Elastic Aircraft for Stability and Control and Flight Control Applications, AFFDL-FGC-TM-72-14. Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, August 1972.

24. Swaim, Robert L. "Aeroelastic Interactions with Flight Control (A Survey Paper)," AIAA Paper 83-2219, 1983.

25. Swaim, Robert L. and Donald G. Fullman, "Prediction of Elastic-Airplane Longitudinal Dynamics from Rigid-Body Aerodynamics," AIAA Journal of Aircraft, 14: 868-873 (September 1977).

26. Taylor, A. S. and D. L. Woodcock. Mathematical Approaches to the Dynamics of Deformable Aircraft, R & M Number 3776. Her Majesty's Stationary Office, London, England, 1976 (AD-B02878).

27. TOTAL User's Manual. Air Force Institute of Technology, Wright-Patterson AFB OH, June 1981.

28. The USAF Stability and Control Digital DATCOM, AFFDL-TR-79-3032, Air Force Flight Dynamics Laboratory Wright-Patterson AFB OH, April 1979.

29. Warren, C. H. E. The Equations of Motion of an Aircraft Embracing its Whole-body and Deformational Degrees of Freedom, RAE-TR-79010. Royal Aircraft Establishment, Farnborough, England, 1979 (AD-A069784).

30. Wilkinson, K., et al. Automated Procedure for Flutter and Strength Analysis and Optimization of Aerospace Vehicles, AFFDL-TR-75-137, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB OH, December 1975.

Vita

John Joseph Cerra II was born on 3 October 1960 in Rome New York. He attended the United States Air Force Academy from June 1978 to June 1982, receiving his Bachelor of Science Degree in Aeronautical Engineering. From July 1982 until May 1985 he worked as a Stability and Control Engineer in the Flight Control Division of The Flight Dynamics Laboratory of the Air Force Wright Aeronautical Laboratories at Wright-Patterson Air Force Base. In May 1985 he entered the Graduate Aeronautical Engineering program at the Air Force Institute of Technology.

Permanent Address: Box 447 Meadow Lane
Apalachin, New York 13732